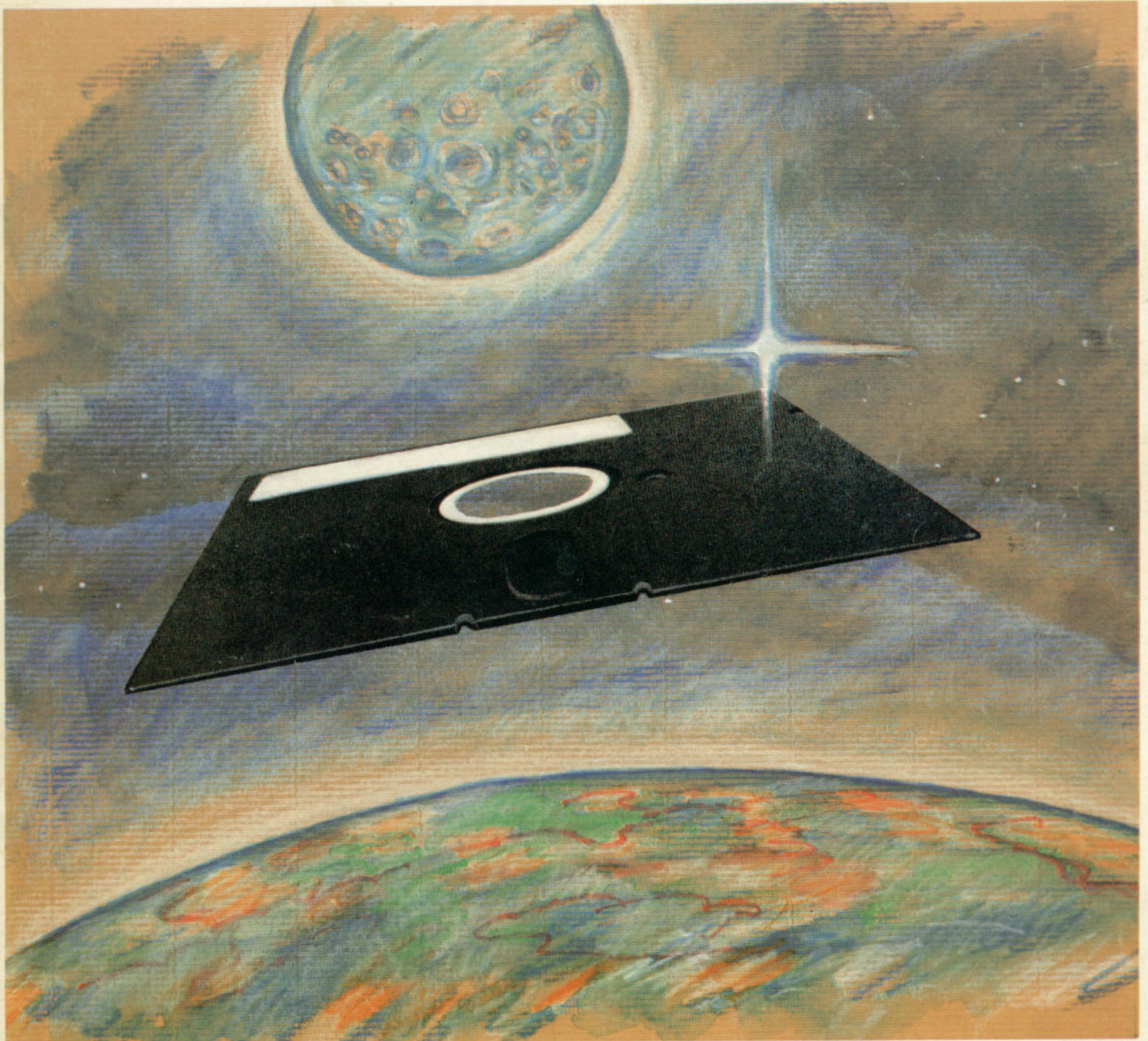
ANO 1 Nº 02
JUNHO 1988
Cz$ 300,00

CPU

A trilha Ø

Descobrindo o número e

Programas residentes

O incrível Hook

CPU

Águia Informática Ltda.
Rua Santa Clara, 98/415
Copacabana
Rio de Janeiro – RJ
CEP 22041
Tel. (021) 257-4402

DIRETOR RESPONSÁVEL
Gonçalo R. F. Murteira

DIRETORIA TÉCNICA
Antônio F. S. Shalders
Carlos E. A. Moreira
André L. F. de Freitas
J. L. Fonseca

REVISÃO DE TEXTO
Laura Maria Pinto

CAPA
José Aguilera

ASSINATURAS
Eduardo Simplício

ADMINISTRAÇÃO
José Newton Barros

Temos a satisfação de apresentar-lhe o segundo número de CPU, onde você irá encontrar o primeiro projeto de hardware e listagens de programas para MSX escritas em outra linguagem sem ser o já conhecido BASIC.

Nossa intenção ao publicar listagens de programas escritos em outras linguagens, é tornar o leitor familiarizado com outros tipos de programação e abrir caminho para a descoberta das facilidades que elas podem oferecer.

O primeiro projeto de hardware resolve um problema bastante comum aos usuários MSX: as fontes de alimentação para drive e data corder. Lendo o artigo, você terá toda as informações necessárias para à montagem, e acreditamos que não irá deixar qualquer dúvida sobre o projeto.

Seguindo nossa linha, que é a de não "esconder o leite", você irá encontrar muitas dicas para ajudá-lo nos seus programas.

A seção destinada aos jogos sofreu um incremento e incluímos, além das dicas de mil vidas, a coluna de últimas novidades e mapas de jogos.

Mais uma vez o convidamos a participar, enviando-nos os seus programas, dicas e sugestões, para que possamos melhorar sempre.

ÍNDICE

INCLUDES NO TURBO PASCAL 4
DESCOBRINDO O NÚMERO 'e' 6
O INCRÍVEL HOOK 8
FONTE PARA DRIVE E DATA CORDER 11
PROGRAMAS RESIDENTES 15
ESTAÇÃO DE TRABALHO 21
A TRILHA Ø 23
IMPRESSÃO DUPLA 24
PROGRAMAÇÃO GRÁFICA EM SCREEN 1 25

SEÇÕES

MÁXIMAS E MÍNIMAS 27
MÚLTIPLA ESCOLHA 28
LIVROS 29
CARTAS 30
MATEMÁGICA 31
JOGOS & HIGH SCORES 32
DICAS 34

Includes no Turbo Pascal

ANTÔNIO F. S. SHALDERS

Os módulos apresentados neste artigo serão de extrema utilidade para quem lida com funções trigonométricas, pois definem um grande número de funções trigonométricas derivadas e hiperbólicas, em dois subprogramas do tipo include.

Um include é um subprograma que é anexado ao programa principal e passa a fazer parte do mesmo.

Para podermos utilizar as novas funções em um determinado programa, devemos incluir, logo após o nome do programa, a linha de inclusão:

```
Exemplo:PROGRAMA TESTE;
        {$I TRG.P}
        {$I HIPER.P}
```

Obviamente, devemos fazer o include que contiver as funções necessárias ao programa em questão.

O include TRG.P é referente às funções trigonométricas derivadas e inversas; o HIPER.P, às hiperbólicas e hiperbólicas inversas.

A função SGN(X), que fornece o sinal referente ao argumento, também foi incluída, pois é utilizada por algumas das novas funções.

Os argumentos deverão ser dados em radianos e os resultados são do tipo real.

Os programas rodam tanto nos sistemas derivados do MS-DOS, como nos derivados do CP/M.

Se o include em questão possuir funções desnecessárias, basta deletá-las com control-Y. Caso contrário, a memória do micro estará sendo desperdiçada e, em caso de geração de um programa auto-executável, este irá tornar-se muito grande.

Se os dois includes forem utilizados simultaneamente, a rotina FUNCTION relativa à função SGN(X) deverá ser deletada de um deles. Caso contrário, haverá erro.

```
FUNCTION SEC(A:REAL):REAL;
    BEGIN
      SEC:=1/COS(A);
    END;
FUNCTION CSC(A:REAL):REAL;
    BEGIN
      CSC:=1/SIN(A);
    END;
FUNCTION TAN(X:REAL):REAL;
    BEGIN
      TAN:=SIN(X)/COS(X);
    END;
FUNCTION CTG(A:REAL):REAL;
    BEGIN
      CTG:=1/TAN(A);
    END;
FUNCTION ARCSIN(A:REAL):REAL;
    BEGIN
      ARCSIN:=ARCTAN(A/SQRT(-A*A+1))
    END;
FUNCTION ARCCOS(A:REAL):REAL;
    BEGIN
      ARCCOS:=-ARCTAN(A/SQRT(-A*A+1))+(PI/2);
    END;

FUNCTION SGN(A:REAL):REAL;
    BEGIN
      IF A < 0 THEN
         SGN:=-1
      ELSE
         SGN:=1
    END;

FUNCTION ARCSEC(X:REAL):REAL;
    BEGIN
      ARCSEC:=ARCTAN(X/SQRT(X*X+1))+SGN(SGN(X)-1)*(PI/2);
    END;

FUNCTION ARCCSC(X:REAL):REAL;
    BEGIN
      ARCCSC:=ARCTAN((X/SQRT(X*X-1))+SGN(X)-1)*(PI/2);
    END;

FUNCTION ARCCTG(X:REAL):REAL;
    BEGIN
      ARCCTG:=ARCTAN(X)+(PI/2);
    END;
```

```
FUNCTION SGN(A:REAL):REAL;
         BEGIN
           IF A < O THEN
              SGN:=-1
           ELSE
              SGN:=1;
         END;
FUNCTION HYPSIN(X:REAL):REAL;
         BEGIN
           HYPSIN:=(EXP(X)-EXP(-X))/2;
         END;
FUNCTION HYPCOS(X:REAL):REAL;
         BEGIN
           HYPCOS:=(EXP(X)+EXP(-X))/2;
         END;
FUNCTION HYPTAN(X:REAL):REAL;
         BEGIN
           HYPTAN:=EXP(-X)/EXP(X)+EXP(-X)*2+1;
         END;
FUNCTION HYPSEC(X:REAL):REAL;
         BEGIN
           HYPSEC:=2/(EXP(X)+EXP(-X));
         END;
FUNCTION HYPACS(X:REAL):REAL;
         BEGIN
           HYPACS:=2/(EXP(X)-EXP(-X));
         END;
FUNCTION HYPCTG(X:REAL):REAL;
         BEGIN
           HYPCTG:=EXP(-X)/(EXP(X)-EXP(-X))*2+1;
         END;
FUNCTION HYPARCSIN(X:REAL):REAL;
         BEGIN
           HYPARCSIN:=LN(X+SQRT(X*X+1));
         END;
FUNCTION HYPARCCOS(X:REAL):REAL;
         BEGIN
           HYPARCCOS:=LN(X+SQRT(X*X-1));
         END;

FUNCTION HYPARCTAN(X:REAL):REAL;
         BEGIN
           HYPARCTAN:=LN((1+X)/(1-X))/2;
         END;

FUNCTION HYPARCSEC(X:REAL):REAL;
         BEGIN
           HYPARCSEC:=LN(SQRT(-X*X+1)+1)/X;
         END;

FUNCTION HYPARCCSC(X:REAL):REAL;
         BEGIN
           HYPARCCSC:=LN(SGN(X)*SQRT(X*X+1)+1)/X;
         END;

FUNCTION HYPARCCTG(X:REAL):REAL;
         BEGIN
           HYPARCCTG:=LN((X+1)/(X-1))/2;
         END;
```

Descobrindo o número 'e'

PIERLUIGI PIAZZI

O MSX não trabalha diretamente com logaritmos decimais (base 10), mas apenas com logaritmos naturais (ou neperianos) de base e.

Para você saber qual o valor aproximado do número e, basta lembrar que:

$$e = e^1 = EXP (1)$$

Portanto, ao digitar:

PRINT EXP (1)

você obterá o valor de **e** que, fornecido em dupla precisão pelo MSX, deverá dar

2.7182818284588

A finalidade deste artigo é mostrar de onde surge um número tão esquisito e que ele não é tão "arbitrário" quanto parece.

Afinal, se podemos escolher a base que o computador usa, por que não optar por um número mais simples?

Para ter uma idéia de que este número não é arbitrário, acompanhe o raciocínio a seguir:

Imagine que você vai ao Banco fazer um investimento que renda 100% ao cabo de um ano. Vamos supor, obviamente, que a gente resida num país normal, destituído de marajás e, portanto, livre de inflação. Receber 100% de juros ao cabo de um ano é um "negocião". Afinal, seu capital fica, no fim do período, multiplicado por 2!

O banco onde você faz o investimento permite a opção de receber 50% por semestre. É a mesma coisa?

Óbvio que não; seu capital é multiplicado por 1,5 no fim do primeiro semestre e, novamente, por 1,5 no fim do segundo semestre. Você terá, no fim do ano, seu capital multiplicado por

1,5 x 1,5 = 2,25

e não apenas 2, como no caso anterior.

Dividir tanto o período quanto a taxa de juros por 2 não produz o mesmo resultado final.

Com os olhos brilhando de cobiça, você solicita uma taxa de juros de 25% (100%/4) por trimestre (ano /4).

Neste caso, seu capital ficará multiplicado por:

1,25 x 1,25 x 1,25 x 1,25

Como os cálculos ficam mais complicados, você começa a usar seu MSX para descobrir o resultado, digitando:

PRINT (1,25) 4

e percebe que seu capital ficará multiplicado por 2,44140625!

Nesta altura, já fazendo um programinha para descobrir de quanto vai enriquecer se o banco pagar 100%/12 ao mês:

PRINT (1 + 1/12)^ 12

Oba! O capital fica multiplicado por 2,6130352902232!

Vislumbrando lucros infinitos, você pede ao gerente para dividir a taxa de juros e o período por um número N, tão grande quanto você queira.

Será que há um limite para tudo isso? Para descobrir, digite o programa da figura 1, que permite calcular de quanto fica multiplicado seu capital, aumentando indefinidamente N (a cada cálculo, vamos multiplicar N por 10 para fazê-lo crescer rapidamente).

Rodando este programa, você deve obter os resultados mostrados na figura 2.

FIGURA 1

```
100 SCREEN 0:KEY OFF
110 N=1
120 PRINT" N";
130 PRINT TAB(21);"(1+1/N)^N":PRINT
140 FOR I=1 TO 14
150 C=(1+1/N)^N
160 PRINT N;TAB(21);
170 PRINT USING"#.##############";C
180 N=N*10
190 NEXT I
200 PRINT:PRINT TAB(18);"e=";EXP(1)
```

```
10                  2.5937424601000
100                 2.7048138294216
1000                2.7169239322549
10000               2.7181459268134
100000              2.7182682371743
1000000             2.7182804693191
10000000            2.7182816925447
100000000           2.7182818148674
1000000000          2.7182818270996
10000000000         2.7182818283229
100000000000        2.7182818284453
1000000000000       2.7182818284575
10000000000000      2.7182818284586

                 e= 2.7182818284588
```

FIGURA 2

Como você notou, o crescimento, mesmo usando uma taxa de juros praticamente instantânea, é limitado e este limite é, justamente, o número "e".

Esta característica de limite de um crescimento instantâneo é que dá ao "e" propriedades tão interessantes a ponto de elegê-lo como um dos números mais importantes da matemática!

Tão importante que o sistema operacional do MSX acha muito mais fácil (e rápido) calcular funções exponenciais e logaritmos usando-o como base.

Se você quiser calcular um logaritmo em outra base, porém, basta lembrar a propriedade:

$$\log_B A = \log_e A / \log_e B$$

Para calcular, por exemplo, o logaritmo de um número A, na base B, no BASIC MSX basta usar esta propriedade como mostra o exemplo da figura 3.

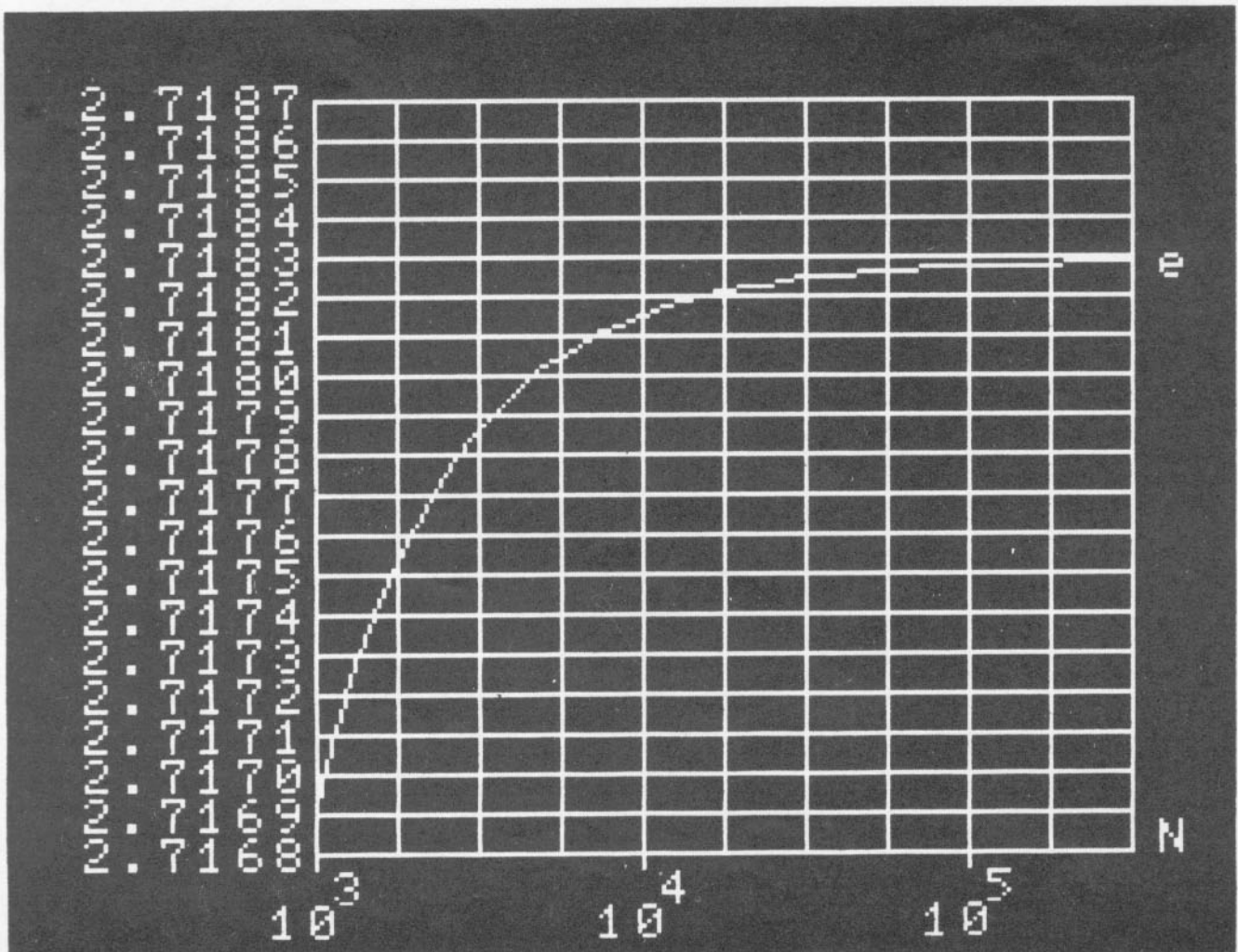
Para finalizar, voltando ao problema dos juros instantâneos, você pode digitar o programa da figura 4 e aprenderá alguma coisa útil, não só em relação ao número "e", mas também à técnica de construção de gráficos em BASIC MSX, como mostra a figura 5.

```
10 DEF FN LD(A,B)=LOG(A)/LOG(B)
20 INPUT "QUAL O NUMERO";A
30 INPUT "QUAL A BASE";B
40 Z= FN LD (A,B)
50 PRINT"O LOGARITMO DE:";A
60 PRINT"NA BASE       :";B
70 PRINT"VALE          :";Z
```

FIGURA 3

```
100 SCREEN 2:Z=174
110 OPEN"GRP:" AS#1:W=10000
120 FOR L=0 TO 19
130 LG=170-8*L
140 PRESET (16,LG-3)
150 PRINT#1,USING"#.####";2.7168+L/W
160 LINE (64,LG)-(234,LG)
170 NEXT L
180 FOR C=0 TO 10
190 CG= 64+17*C
200 LINE(CG,170)-(CG,18)
210 IF C\4 =C/4 THEN PRESET(CG-4,Z):
    PRINT#1,3+C/4:PRESET(CG-17,181):
    PRINT#1,10:LINE(CG,173)-(CG,170)
220 NEXT C
230 PRESET(240,164):PRINT#1,"N"
240 PSET(64,170)
250 FOR I=3 TO 5.5 STEP .0625
260 E=(1+(1/(10^I)))^(10^I)
270 X=64+68*(I-3)
280 Y=170-((E-2.7168)*8*10000)
290 LINE-(X,Y)
300 NEXT I
310 PRESET(240,48):PRINT#1,"e"
320 GOTO 320
```

FIGURA 4

FIGURA 5

O Incrível Hook

ANDRÉ L. F. DE FREITAS

Os micros MSX possuem uma área de memória RAM reservada para uso do sistema. Lá são encontradas variáveis do sistema utilizadas pelo BIOS (Basic Input Output System) e pelo interpretador BASIC. Não confundir este tipo de variável com as variáveis que podemos ter no BASIC. As variáveis de sistema tem posição fixa na memória e são de acesso exclusivo do sistema MSX, podendo o usuário lê-las ou escrevê-las somente através de PEEKs e POKEs ou utilizando linguagem de máquina, mas sabendo muito bem o que faz, pois o uso indevido destas variáveis pode confundir todo o sistema, ocasionando resultados imprevisíveis. Após a área de variáveis de sistema, temos uma seção de memória, mais precisamente entre os endereços &HFD9A e &HFFC9, que,se observarmos com um monitor assembly, perceberemos estar preenchida com bytes &HC9 (instrução RET do assembly Z80). Dividindo esta região em grupos de 5 bytes, podemos chamar cada grupo destes de HOOK (vou usar aqui a palavra da língua inglesa, pois as traduções geralmente se tornam sem sentido). A princípio pode parecer que área da memória preenchida com RETs não tem importância nenhuma, mas, no caso do MSX, estes Hooks são de grande utilidade.

O BIOS e o BASIC MSX chamam, em alguns pontos de sua execução, endereços nesta área de memória. Lá, encontram um RET e retornam à origem sem executar nada. É o caso dos comandos de Disk BASIC e a própria rotina de leitura de teclado. Podemos, então, colocar nestes pequenos Hooks, também, pequenas rotinas para chamar outras áreas da memória, criando um salto a partir de rotinas do sistema que estavam em ROM e, portanto, não poderiam ser modificadas, conseguindo, desta forma modificar o próprio sistema operacional.

Cinco bytes são mais do que suficientes para conter uma instrução CALL, seu endereço correspondente, e um RET no final para a volta ao sistema operacional. É assim que funcionam os comandos do Disk BASIC. Desta forma, podemos ter extensões do BASIC ou do BIOS sem a necessidade de alteração na ROM do equipamento.

Como exemplo de utilização dos Hooks, os POKEs que forneço a seguir vão alterar um Hook que é chamado pela rotina de impressão de caracteres no vídeo, fazendo uma chamada à rotina do BEEP do BASIC MSX a cada caracter impresso no vídeo. A ordem em que os POKEs são dados, do último ao primeiro endereço, é necessária para o Hook não ficar com os bytes &HC3 &HC9 &HC9, após o primeiro POKE, o que causaria um salto para &HC9C9 (CALL C9C9H), resultando na perda de controle do sistema. Os POKEs são os seguintes:

```
POKE &HFDA6, 0
POKE &HFDA5, &HCO
POKE &HFDA4, &HCD
```

Experimente, agora, digitar algo, entrar um programa em BASIC e dar um LIST depois, ou qualquer coisa que produza caracteres no vídeo. Repare no BEEP a cada impressão e na velocidade com que os caracteres são apresentados, que é bem menor que a normal devido à chamada da rotina do BEEP. Para voltar ao normal, basta um POKE &HFDA4, &HC9, que restaura o Hook ao original. É esta a utilidade dos Hooks: permitir o desvio de rotinas do sistema para outras rotinas criadas por você mesmo ou já existentes no micro. É uma forma simples de se expandir o sistema MSX por software, sem nenhuma alteração no equipamento.

No desenvolvimento do MSX, foram reservados 5 bytes para cada Hook, sendo estes criados de forma a aproveitar as rotinas mais fundamentais do funcionamento do MSX, permitindo futuras alterações. O porquê dos cinco bytes é o seguinte: permitir a inclusão no Hook de uma chamada a rotinas em qualquer slot, feito da forma descrita abaixo:

```
REST 30H                                  1 byte)
DEFB Identificador de slot               (1 byte)
DEFW Endereço de execução da
       rotina                            (2 bytes)
RET                                       1 byte)
```

A instrução RST 30H desvia a execução para o endereço &H0030 da memória. Lá existe uma rotina que acessa um endereço em um slot especificado (bytes após a instrução RST 30H), executando o que for encontrado e retornando a seguir. É esta a forma de execução de um comando do Disk BASIC. A interface doDisk Drive possui uma ROM adicional, que expande o sistema MSX para os comandos Disk BASIC, que são chamados desta forma. Não entrarei em mais detalhes sobre esta chamada, ficando para uma outra ocasião um melhor tratamento deste método.

Encerro este artigo com uma lista dos Hooks existentes no MSX, seus endereços, e o endereço da ROM de onde são chamados. Espero que tenham grande utilidade para todos. Com o conhecimento destes Hooks o leitor não será mais um leigo quando, eventualmente, eu os mencionar ou utilizar em futuros artigos.

Ender.	Hook	Endereco de onde e chamado e rotina que chama o Hook
FD9AH	HKEYI	0C4AH Interrupcao de teclado
FD9FH	HTIMI	0C53H Interrupcao de teclado
FDA4H	HCHPU	08C0H Rotina CHPUT (impr. caracteres)
FDA9H	HDSPC	09E6H Mostra cursor
FDAEH	HERAC	0A33H Apaga cursor
FDB3H	HDSPF	0B2BH Rotina DSPFNK (mostra funcoes)
FDB8H	HERAF	0B15H Rotina ERAFNK (apaga funcoes)
FDBDH	HTOTE	0842H Rotina TOTEXT (tela de texto)
FDC2H	HCHGE	10CEH Rotina CHGET (le caracter)
FDC7H	HINIP	071EH Copia tabela de caract. p/ VDP
FDCCH	HKEYC	1025H Decodificador de teclado
FDD1H	HKEYA	0F10H Decodificador de teclado
FDD6H	HNMI	1398H Interrupcoes nao mascaraveis
FDDBH	HPINL	23BFH Rotina PINL (input)
FDE0H	HQINL	23CCH Rotina QINL (input)
FDE5H	HINLI	23D5H Rotina INLIN (input)
FDEAH	HONGO	7810H Instrucao ON DEVICE GOSUB
FDEFH	HDSKO	7C16H Instrucao DSKO$
FDF4H	HSETS	7C1BH Instrucao SET
FDF9H	HNAME	7C20H Instrucao NAME
FDFEH	HKILL	7C25H Instrucao KILL
FE03H	HIPL	7C2AH Instrucao IPL
FE08H	HCOPY	7C2FH Instrucao COPY
FE0DH	HCMD	7C39H Instrucao CMD
FE12H	HDSKF	7C3EH Instrucao DSKF
FE17H	HDSKI	7C43H Instrucao DSKI$
FE1CH	HATTR	7C48H Instrucao ATTR$
FE21H	HLSET	7C4DH Instrucao LSET
FE26H	HRSET	7C52H Instrucao RSET
FE2BH	HFIEL	7C57H Instrucao FIELD
FE30H	HMKI$	7C5CH Instrucao MKI$
FE35H	HMKS$	7C61H Instrucao MKS$
FE3AH	HMKD$	7C16H Instrucao MKD$
FE3FH	HCVI	7C66H Instrucao CVI
FE44H	HCVS	7C6BH Instrucao CVS
FE49H	HCVD	7C70H Instrucao CVD
FE4EH	HGETP	6A93H Localiza FCB
FE53H	HSETF	6AB3H Localiza FCB
FE58H	HNOFO	6AF6H Instrucao OPEN
FE5DH	HNULO	6B0FH Instrucao OPEN
FE62H	HNTFL	6B3BH Fecha buffer I/O 0
FE67H	HMERG	6B63H Instrucao MERGE/LOAD
FE6CH	HSAVE	6BA6H Instrucao SAVE
FE71H	HBINS	6BCEH Instrucao SAVE
FE76H	HBINL	6BD4H Instrucao MERGE/LOAD
FE7BH	HFILE	6C2FH Instrucao FILES
FE80H	HDGET	6C3BH Instrucao GET/PUT
FE85H	HFILO	6C51H Saida Sequencial
FE8AH	HINDS	6C79H Entrada Sequencial
FE8FH	HRSLF	6CD8H Instrucao INPUT$
FE94H	HSAVD	6D03H Instrucao LOC
		6D25H Instrucao EOF
		6D39H Instrucao FPOS
		6D14H Instrucao LOF
FE99H	HLOC	6D0FH Instrucao LOC
FE9EH	HLOF	6D20H Instrucao LOF
FEA3H	HEOF	6D33H Instrucao EOF
FEA8H	HFPOS	6D43H Instrucao FPOS
FEADH	HBAKU	6E36H Instrucao LINE INPUT#
FEB2H	HPARD	6F15H Analise de dispositivos
FEB7H	HNODE	6F33H Analise de dispositivos
FEBCH	HPOSD	6F37H Analise de dispositivos
FEC1H	HDEVN	HOOK NAO USADO
FEC6H	HGEND	6F8FH Funcoes de I/O
FECBH	HRUNC	629AH Run/clear
FED0H	HCLEA	62A1H Run/clear
FED5H	HLOPD	62AFH Run/clear
FEDAH	HSTKE	62F0H Inicializacao do STACK
FEDFH	HISFL	145FH Rotina ISFLIO (direcao de buffer)
FEE4H	HOUTD	1B46H Rotina OUTDO (saida em dispositivo
FEE9H	HCRDO	7328H Saida de CR e LF em OUTDO
FEEEH	HDSKC	7374H Entrada de linhas em buffer
FEF3H	HDOGR	593CH Tracado de linhas
FEF8H	HPRGE	4039H Fim de programa
FEFDH	HERRP	40DCH Rotina de erros
FF02H	HERRF	40FDH Rotina de erros
FF07H	HREAD	4128H OK na execucao
FF0CH	HMAIN	4134H Execucao de programas
FF11H	HDIRD	41A8H Execucao de comandos
FF16H	HFINI	4237H Fim de execucao
FF1BH	HFINE	4247H Fim de execucao
FF20H	HCRUN	42B9H Conversao em TOKENS
FF25H	HCRUS	4353H Conversao em TOKENS
FF2AH	HISRE	437CH Conversao em TOKENS
FF2FH	HNTFN	43A4H Conversao em TOKENS
FF34H	HNOTR	44EBH Conversao em TOKENS
FF39H	HSNGF	45D1H Instrucao FOR
FF3EH	HNEWS	4601H NEW em execucao
FF43H	HGONE	4646H Execucao de programas
FF48H	HCHRG	4666H Rotina CHRGTR (pega caracter)
FF4DH	HRETU	4821H Instrucao RETURN
FF52H	HPRTF	4A5EH Instrucao PRINT
FF57H	HCOMP	4A94H Instrucao PRINT
FF5CH	HFINP	4AFFH Instrucao PRINT
FF61H	HTRMN	4B4DH Erro em READ/INPUT
FF66H	HFPME	4C6DH Avaliador de expressoes
FF6BH	HNTPL	4CA6H Avaliador de expressoes
FF70H	HEVAL	4DD9H Avaliador de fatores
FF75H	HOKNO	4F2CH Avaliador de fatores
FF7AH	HFING	4F3EH Avaliador de fatores
FF7FH	HISMI	51C3H Execucao de programas
FF84H	HWIDT	51CCH Instrucao WIDTH
FF89H	HLIST	522EH Instrucao LIST
FF8EH	HBUFL	532DH Conversao de TOKENS em texto
FF93H	HFRQI	543FH Conversao para inteiros
FF98H	HSCNE	5514H Numero de linhas p/ ponteiros
FF9DH	HFRET	67EEH Limpa STRINGS
FFA2H	HPTRG	5EA9H Procura de variaveis
FFA7H	HPHYD	148AH Rotina PHIDIO (disco)
FFACH	HFORM	148EH Rotina FORMAT (disco)
FFB1H	HERRO	406FH Rotina de erros
FFB6H	HLPTO	085DH Rotina LPTOUT (impressora)
FFBBH	HLPTS	0884H Rotina LPTTST (impressora)
FFC0H	HSCRE	79CCH Instrucao SCREEN
FFC5H	HPLAY	73E5H Instrucao PLAY

Fonte para drive e data corder

CARLOS E. A. MOREIRA

Como foi prometido no editorial do primeiro número de nossa revista além de software, seriam publicados, também, projetos de hardware, e, como promessa é dívida, reservamos aqui um espaço para que possamos mostrar alguns projetos que, além der serem relativamente fáceis de ser confeccionados, mostrar-se-ão de uma indiscutível utilidade.

Por hora, penso eu, que o que tem dado mais dor de cabeça aos usuários do MSX são as fontes que se destinam à alimentação do "disk driver" e, também, para os que possuem a versão 1.0 do EXPERT ou do HOT BIT no que diz respeito à alimentação do DATACORDER, visto que nestes dois não há uma saída para a alimentação do cassete. O que é exigido destas fontes é precisão na regulagem, ou seja, que a tensão se mantenha em um nível constante ao longo do tempo. No caso de fonte para o "disk driver", utilizamos o integrado CA723 da RCA, um circuito confiável a este tipo de aplicação. Para os mais curiosos, este integrado, CA723, é o regulador usado, também na fonte de alimentação do HOT BIT. Para a fonte do DATACORDER usamos um outro integrado, o uA7806, tão confiável quanto o anterior, porém mais adequado neste caso.

Então vamos ao que interessa realmente que são as fontes. Na FIGURA 1 podemos observar o esquema completo da fonte que se destina ao "disk driver". Como sabemos, o "disk driver" necessita de duas tensões de operação: uma de 5 volts, que irá alimentar os circuitos TTL, e uma outra de 12 volts, que se prestará a alimentar os motores de passo. Assim sendo, foram utilizados dois circuitos integrados do tipo uA723, cada um controlando uma das duas tensões. Para melhor compreensão do circuito, resolvi "quebrá-lo" em partes e tentar descrevê-las separadamente.

Na FIGURA 2 está o módulo que se destina a fazer a regulagem de 12 volts. Nela temos, como já dissemos antes, o integrado CA723 e o transistor TIP3055 com o objetivo de podermos ter na saída regulada uma corrente sensivelmente maior que o CA723 pode nos fornecer, que é da ordem de 150 miliampéres, no máximo. Aqui, estamos exigindo uma corrente de, pelo menos, 1,5 ampéres. O divisor resistivo 2K2/3K9 nos dá uma tensão próxima dos 12 volts. Assim, foi colocado em série com esses, dois resistores um trimpot de 470 ohms. Com isso, podemos fazer um ajuste fino de tensão e colocarmos a saída exatamente em 12 volts, ou bastante próximo disso.

Na FIGURA 3 está o módulo regulador de 5 volts, que muda pouca coisa em comparação com o circuito da FIGURA 2. Aqui, novamente, temos um par de resistores fazendo um divisor resistivo (1K5/4K7) que nos fornece uma tensão próxima do exigido e, também, um trimpot (470R) em série com estes resistores para o ajuste fino. O transistor (TIP32) tem exstamente o mesmo objetivo que no circuito da FIGURA 2.

Na FIGURA 4 sugerimos um "lay out" da placa impressa que poderá ser usada na montagem. O transformador deverá ser um de 110 ou 220 volts no primário, dependendo, é claro, da tensão da rede, e de 12 volts no secundário que possa, também, fornecer uma corrente de no mínimo 2 ampéres.

A fonte para o DATACORDER é sensivelmente mais simples. Aqui usamos um outro tipo de regulador integrado, o uA7806, que já nos fornece uma tensão fixa (6.0 volts), o que quer dizer que, neste caso, não necessitamos de nenhum ajuste. Desta maneira, terminada a montagem do circuito, este estará pronto para o "batente", enquanto que no circuito anterior, terminada a montagem, este necessitará de um ajuste prévio, o que será conseguido com o auxílio de um voltímetro. O esquema desta fonte poderá ser observado na FIGURA 5.

Na FIGURA 6 tem-se uma sugestão do "lay out" do circuito impresso usado na montagem.

Um detalhe importante é que tanto o integrado uA7806 quanto os transistores TIP3055 e TIP32 necessitam, cada um deles, de dissipadores de calor. Podemos notar que todos esses componentes já são preparados para receber esses dissipadores, e que, nestes projetos aqui apresentados, posso garantir que são absolutamente INDISPENSÁVEIS, já que, através destes componentes, circulará uma corrente sensivelmente elevada.

Uma sugestão para facilitar a montagem do circuito da FIGURA 1 é usar soquetes ao invés de soldarmos diretamente o C.I. CA723 na placa impressa. Isso nos dá uma facilidade a mais, pois, para soldarmos este soquete, é preciso um cuidado extremamente menor do que se formos soldar o integrado diretamente sobre a placa.

Terminada a montagem e conferidas todas as ligações, chega a hora dos ajustes. Nos pontos descritos como, saídas de 12 e 5 volts, respectivamente, introduzimos um voltímetro. Este, provavelmente, dará leituras próximas das tensões requeridas. Assim, agindo sobre o trimpot, em cada um dos casos, podemos, afinar as duas tensões. Todos os circuitos apresentados aqui foram submetidos aos testes e apresentaram características ótimas em relação ao esperado.

Todos os resistores são de 1/8 de WATT, a não ser que algo indique o contrário.

O C.I. CA723 DEVERÁ ser do tipo DIL-14, com encapsulamento plástico, e nomenclaturas como, por exemplo, uA723 ou LM723. Isso porque a notação varia de fabricante para fabricante.

Todos os componentes são facilmente encontrados no mercado de componentes eletrônicos.

Qualquer dúvida poderá ser desfeita, bastando, para isso, escrever para ÁGUIA INFORMÁTICA. Terei imenso prazer em respondê-las, caso ocorram, é claro. Até a próxima.

LISTA DE COMPONENTES

1. Para a fonte do "Disk Driver"

- Circuito integrado CA723 (2)

- Resistores:
  68R (azul cinza preto)
  1K5 (marrom verde vermelho)
  2K2 (vermelho vermelho vermelho)
  3K9 (laranja branco vermelho)
  4K7 (amarelo violeta vermelho)
  47OR (trimpot) (2)
  0.22 OHM X 2 WATTS (resistor de fio)
  0.33 OHM X 2 WATTS (resistor de fio)

- Capacitores:
  1 KpF (cerâmico) (2)
  100 uF X 16 volts (eletrolítico) (2)
  100 KpF (poliéster)
  2200 uF X 25 volts (eletrolítico)

- Transistores:
  TIP32 (pnp)
  TIP3055 (npn)

- Diodos
  SK2/02 (2)

- Transformador 110 ou 220 volts no primário e 12 volts, no secundário, com no mínimo 2 Ampéres de corrente.

2. Para a fonte do DATACORDER

- Circuito integrado ua7806

- Resistores:
  5K6 (verde azul vermelho)

- Capacitores:
  100 pF (cerâmico)
  330 KpF (poliéster)
  1 uF X 16 volts (tântalo)
  2200 uF X 25 volts (eletrolítico)

- Diodos:
  IN4007

Transformador 110 ou 220 volts no primário e 9 volts no secundário que forneça uma corrente de 500 miliampéres.

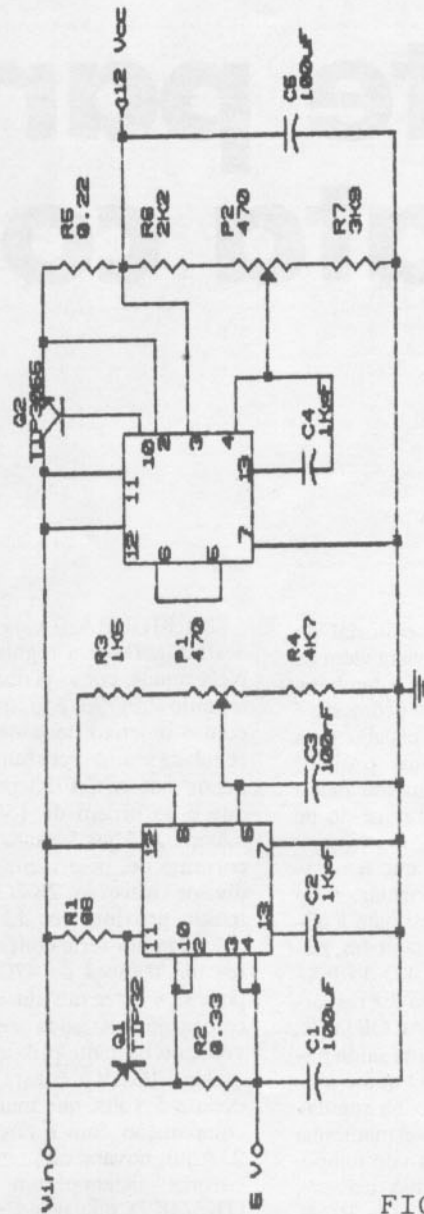
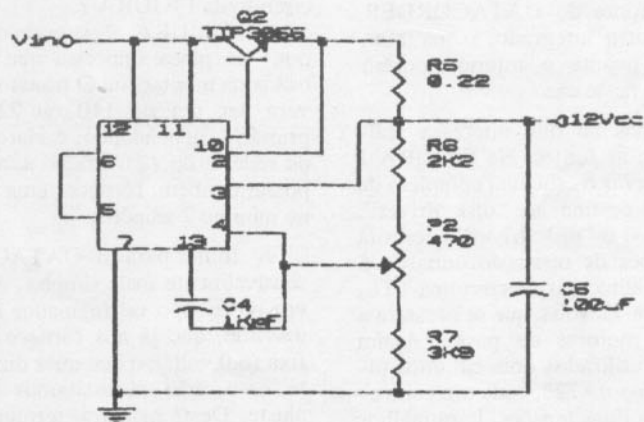



FIGURA 1

FIGURA 2

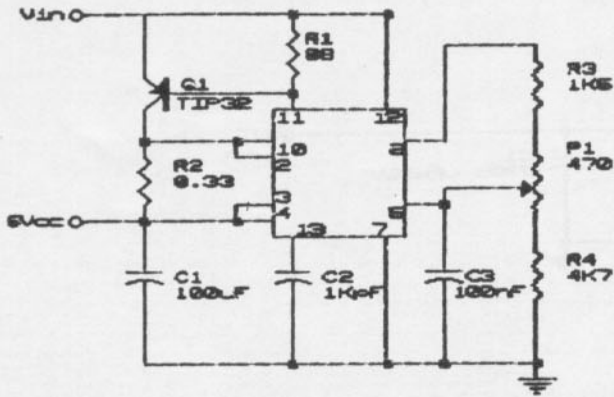



FIGURA 3

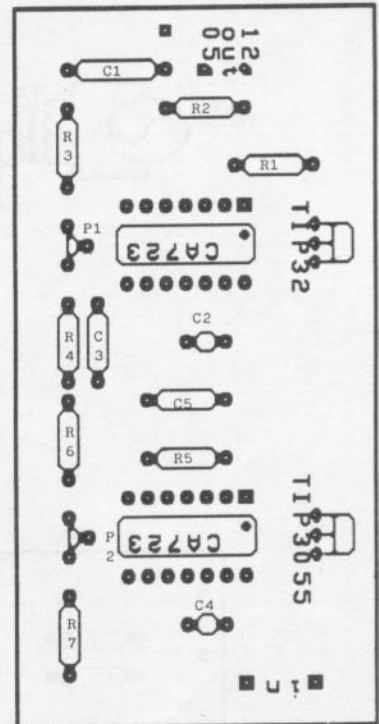



FIGURA 4

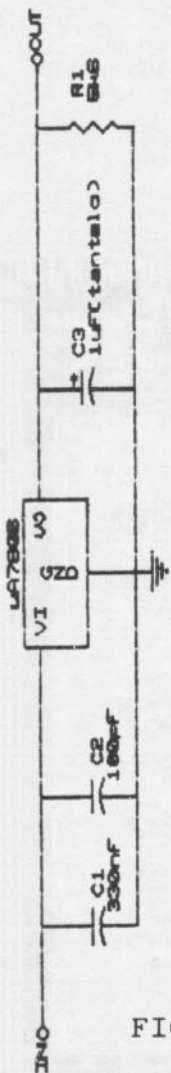



FIGURA 6

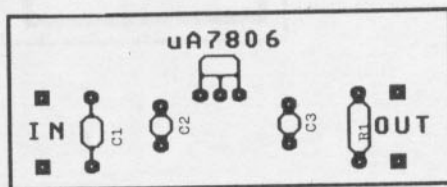
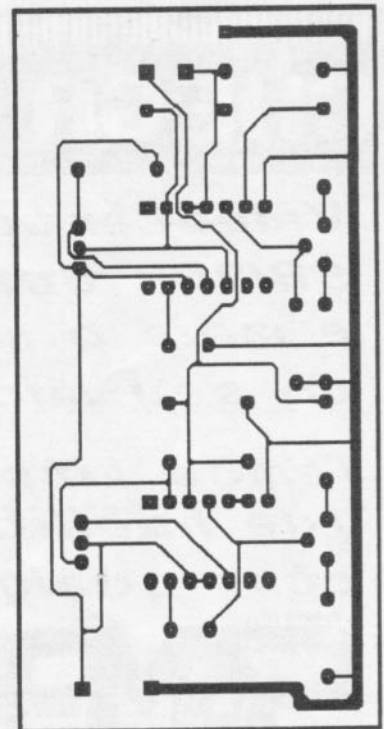
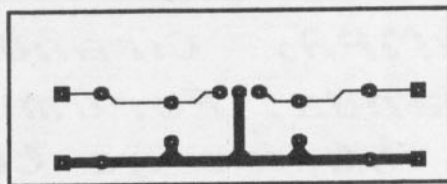
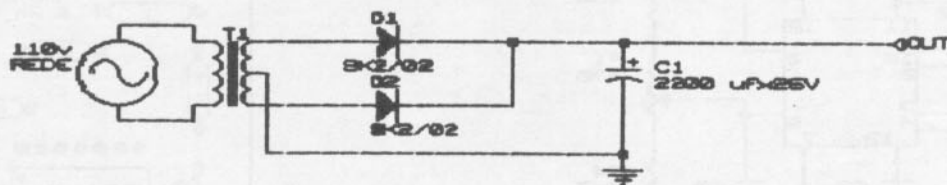
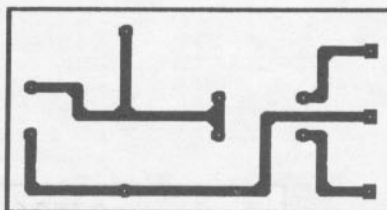
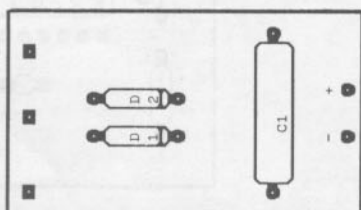



FIGURA 5

ESQUEMA PARA RETIFICAÇÃO

Programas Residentes

ANDRÉ L. F. DE FREITAS

Em algumas outras linhas de microcomputadores muito comuns no mercado internacional, sendo estes fabricados também aqui no Brasil, encontramos um tipo de software muito interessante. São programas que, ao serem carregados, aparentemente não exercem função alguma; retornam ao sistema operacional apresentando, somente, uma mensagem do tipo: Pressione as teclas X e Y simultaneamente para executar.

De fato, são programas extremamente interessantes, pois ficam na memória do micro sem aparecer ao usuário e, ao serem chamados através de um conjunto de teclas, interrompem qualquer processo que o micro esteja executando e assumem o controle, oferecendo ao usuário recursos como redefinição de teclado, calculadora, agenda, etc.

Mais interessante é saber como estes programas ficam "pendurados" no sistema sem atrapalhar o processamento normal do micro.

O princípio é o seguinte: os processadores comumente encontrados em microcomputadores possuem modos em que podem ser interrompidos, desviando o processo corrente para uma outra rotina, executando a esma e retornando ao processo anterior. No caso do MSX, isto é feito pelo hardware 60 vezes por segundo, gerando a interrupção de leitura de teclado. É desta forma que o teclado do MSX é varrido e o código da tecla encontrado. Baseado em interrupções deste tipo, surgiu o software "terminate and stay resident", que podemos chamar de softawre residente.

O programa, inicialmente, é carregado para a memória e uma pequena rotina é colocada estrategicamente dentro da rotina que é chamada pela interrupção. Esta rotina fica acompanhando a leitura de teclado, aguardando o determinado conjunto de teclas. Este conjunto, ao ser detectado, gera um salto para o programa na memória e este se encarrega de guardar informações a respeito do processo que o micro estava executando. Após esta operação, o programa residente é normalmente executado e, ao ser terminado, restaura todo o processo anteriormente executado pelo micro, prosseguindo do ponto onde parou.

O objetivo deste artigo é mostrar ao usuário MSX como fazer o mesmo com seu micro, aproveitando aquela memória RAM que não é normalmente utilizada pelo MSX. Atente para o fato de não ser fornecido um programa residente como os encontrados no mercado para outros micros, mas somente um pequeno programa que irá limpar a tela e imprimir uma mensagem. O que eu quero é mostrar o processo pelo qual o programa residente é instalado e, futuramente, chamado pelo teclado.

A listagem 1 contém o programa residente e seu instalador, que é encontrado a partir de seu endereço inicial de exeução até o seu final.

O instalador vai transportar o programa residente do local one foi carregado para um slot de RAM livre no MSX. A seguir, transportará para uma área livre no final da memória RAM (temos uma área não utilizada entre os endereços &HFFCA e &HFFFF) uma rotina que testará se a tecla [ESC] foi pressionada e decidirá, caso sim ou não, respectivamente, se chama o programa residente ou retorna sem nada executar.

O terceiro passo será alterar o HOOK do sistema correspondente à leitura de teclado (Hook HKEYG – interrupção de teclado) para um desvio para aquele programa instalado no fim da RAM. Está pronta a instalação do programa residente. Uma mensagem será apresentada instruindo o usuário a como utilizar o programa residente. Para aqueles que não sabem ainda o que são os Hooks, sugiro procurar informações em outro artigo de minha autoria publicado neste mesmo número de CPU.

Muito cuidado para não digitar códigos errados, pois isto pode ocasionar a perda de controle do sistema e ser necessário pressionar o famigerado botão de RESET do micro para retornar ao sistema, perdendo-se todo o trabalho feito até então. É uma boa idéia salvar o programa, antes, para não perder alguns fios de cabelo (chato, não?).

De posse do conhecimento de todas as informações encontradas neste e no outro artigo, fica ao usuário a tarefa de criar o seu programa residente e substituir o exemplo dado pelo próprio. Experimente, também, tentar alterar a rotina de teste de teclado para um outro código de teclas a acionar o programa residente.

Outra coisa a lembrar é não chamar o programa residente se você estiver trabalhando fora do SCREEN 0, pois este programa residente não prevê a restauração de outro modo de tela a não ser o modo 0.

Futuramente, publicarei programas residentes que poderão ser substituídos pelo fornecido por este artigo, como, por exemplo, uma calculadora ou uma tabela de códigos ASCII, que serão muito úteis instalados no micro durante o desenvolvimento de outros programas, mesmo que você só programe em BASIC. Aguarde outros artigos desta série.

```
1000                    ; REVISTA CPU
1010                    ; JUNHO 88
1020                    ; PROGRAMAS RESIDENTES
1030                    ; A.L. FREITAS
1040                    ;
1050                    ; ROTINAS DO BIOS E
1060                    ; VARIAVEIS DO SISTEMA
1070                    ;
1080    0059            LDIRMV:EQU  0059H          ; COPIA VRAM-RAM
1090    005C            LDIRVM:EQU  005CH          ; COPIA RAM-VRAM
1100    006C            INITXT:EQU  006CH          ; ENTRA EM TELA DE TEXTO
1110    009F            CHGET: EQU  009FH          ; LE CARACTER
1120    00A2            CHPUT: EQU  00A2H          ; IMPRIME CARACTER
1130    00C3            CLS:   EQU  00C3H          ; LIMPA TELA
1140    00C6            POSIT: EQU  00C6H          ; POSICIONA CURSOR
1150    0156            KILBUF:EQU  0156H          ; LIMPA BUFFER TECLADO
1160    F3DC            CSRY:  EQU  0F3DCH         ; VAR. END. CURSOR
1170    FDD1            HKEYA: EQU  0FDD1H         ; HOOK LEITURA DE TECLADO
1180                    ;
1190                    ; PROGRAMA RESIDENTE
1200                    ;
1210                           ORG  0C000H
1220                    ;
1230     C000 C5        RESIDE:PUSH BC             ; SALVA REG.
1240     C001 D5               PUSH DE
1250     C002 E5               PUSH HL
1260     C003 2ADCF3           LD   HL,(CSRY)      ; SALVA CURSOR
1270     C006 E5               PUSH HL
1280     C007 210000           LD   HL,0
1290     C00A 110050           LD   DE,5000H
1300     C00D 01BF03           LD   BC,03BFH
1310     C010 CD5900           CALL LDIRMV         ; COPIA VRAM-RAM
1320     C013 CDC300           CALL CLS
1330     C016 210401           LD   HL,0104H
1340     C019 22DCF3           LD   (CSRY),HL      ; POSICIONA CURSOR
1350     C01C 214140           LD   HL,MENSAG-8000H
1360     C01F 7E        LOOP:  LD   A,(HL)         ; IMPRIME MENSAGEM
1370     C020 FE00             CP   0
1380     C022 2806             JR   Z,ENDL
1390     C024 CDA200           CALL CHPUT
1400     C027 23               INC  HL
1410     C028 18F5             JR   LOOP
1420     C02A CD9F00    ENDL:  CALL CHGET          ; AGUARDA TECLA
1430     C02D 210050           LD   HL,5000H
1440     C030 110000           LD   DE,0
1450     C033 01BF03           LD   BC,03BFH
1460     C036 CD5C00           CALL LDIRVM         ; RESTAURA TELA
1470     C039 E1               POP  HL
1480     C03A 22DCF3           LD   (CSRY),HL      ; RESTAURA POSIC. CURSOR
1490     C03D E1               POP  HL             ; RESTAURA REG.
1500     C03E D1               POP  DE
1510     C03F C1               POP  BC
1520     C040 C9               RET
1530                    ;
```

```
1540  C041 2A2A2A2A MENSAG:DEFM '********************'
      C045 2A2A2A2A
      C049 2A2A2A2A
      C04D 2A2A2A2A
      C051 2A2A2A2A
1550  C055 2A2A2A2A        DEFM '********************',13,10,10,9,9
      C059 2A2A2A2A
      C05D 2A2A2A2A
      C061 2A2A2A2A
      C065 2A2A2A2A
      C069 0D0A0A09
      C06D 09
1560  C06E 52455649        DEFM 'REVISTA CPU',13,10,10,9,9
      C072 53544120
      C076 4350550D
      C07A 0A0A0909
1570  C07E 4A554E48        DEFM 'JUNHO 88',13,10,10,9,9
      C082 4F203838
      C086 0D0A0A09
      C08A 09
1580  C08B 50524F47        DEFM 'PROGRAMAS RESIDENTES',13,10,10,9,9
      C08F 52414D41
      C093 53205245
      C097 53494445
      C09B 4E544553
      C09F 0D0A0A09
      C0A3 09
1590  C0A4 412E4C2E        DEFM 'A.L. FREITAS',13,10,10,10,9,9
      C0A8 20465245
      C0AC 49544153
      C0B0 0D0A0A0A
      C0B4 0909
1600  C0B6 50524553        DEFM 'PRESSIONE UMA TECLA',13,10,10,9,9
      C0BA 53494F4E
      C0BE 4520554D
      C0C2 41205445
      C0C6 434C410D
      C0CA 0A0A0909
1610  C0CE 50415241        DEFM 'PARA SAIR.',13,10,10,10
      C0D2 20534149
      C0D6 522E0D0A
      C0DA 0A0A
1620  C0DC 2A2A2A2A        DEFM '********************'
      C0E0 2A2A2A2A
      C0E4 2A2A2A2A
      C0E8 2A2A2A2A
      C0EC 2A2A2A2A
1630  C0F0 2A2A2A2A        DEFM '********************',13,10,0
      C0F4 2A2A2A2A
      C0F8 2A2A2A2A
      C0FC 2A2A2A2A
      C100 2A2A2A2A
      C104 0D0A00
1640  C107          ENDR:  EQU  $
1650                ;
1660                ; PROGRAMA INSTALADOR
1670                ; ENTRADA: &HD000
1680                ;
```

```
1690                  ;
1700                        ORG  0D000H
1710   D000 F5        INSTAL:PUSH AF              ; SALVA REG.
1720   D001 C5              PUSH BC
1730   D002 D5              PUSH DE
1740   D003 E5              PUSH HL
1750   D004 CD6C00          CALL INITXT           ; INICIAL. TEXTO
1760   D007 21A2D0          LD   HL,EXCT          ; TRANSF. PROG. EXECUTOR
1770   D00A 11CAFF          LD   DE,0FFCAH        ; PARA AREA LIVRE
1780   D00D 012200          LD   BC,ENDX-EXCT     ; NA RAM
1790   D010 EDB0            LDIR
1800   D012 DBA8            IN   A,(0A8H)         ; TRANSFERE PROG. RESIDENTE
1810   D014 32FDFF          LD   (0FFFDH),A       ; P/ PAGINA LIVRE NA RAM
1820   D017 210040          LD   HL,4000H         ; (MAIORES DETALHES VER ARTIGO
1830   D01A 0603            LD   B,3              ;  SLOTS E EXPANSOES EM CPU
1840   D01C DBA8      LOOP1: IN  A,(0A8H)         ;  NUMERO 1)
1850   D01E C604            ADD  A,4
1860   D020 32FEFF          LD   (0FFFEH),A
1870   D023 D3A8            OUT  (0A8H),A
1880   D025 3EAA            LD   A,0AAH
1890   D027 77              LD   (HL),A
1900   D028 56              LD   D,(HL)
1910   D029 2F              CPL
1920   D02A 77              LD   (HL),A
1930   D02B 5E              LD   E,(HL)
1940   D02C 7A              LD   A,D
1950   D02D 83              ADD  A,E
1960   D02E FEFF            CP   0FFH
1970   D030 280C            JR   Z,ACHOU
1980   D032 10E8            DJNZ LOOP1
1990   D034 3AFDFF          LD   A,(0FFFDH)
2000   D037 D3A8            OUT  (0A8H),A
2010   D039 E1              POP  HL
2020   D03A D1              POP  DE
2030   D03B C1              POP  BC
2040   D03C F1              POP  AF
2050   D03D C9              RET
2060   D03E 2100C0    ACHOU: LD  HL,RESIDE
2070   D041 110040          LD   DE,4000H
2080   D044 010701          LD   BC,ENDR-RESIDE
2090   D047 EDB0            LDIR
2100   D049 3AFDFF          LD   A,(0FFFDH)
2110   D04C D3A8            OUT  (0A8H),A
2120   D04E 21CAFF          LD   HL,0FFCAH        ; END. DE EXECUTOR REALOCADO
2130   D051 2202FD          LD   (HKEYA+1),HL     ; COPIA P/ O HOOK
2140   D054 3ECD            LD   A,0CDH           ; CODIGO DE CALL
2150   D056 3201FD          LD   (HKEYA),A        ; COPIA P/ O HOOK
2160   D059 216CD0          LD   HL,MENSG
2170   D05C 7E        LOOP2: LD  A,(HL)           ; IMPRIME MENSAGEM
2180   D05D FE00            CP   0
2190   D05F 2806            JR   Z,ENDL2
2200   D061 CDA200          CALL CHPUT
2210   D064 23              INC  HL
2220   D065 18F5            JR   LOOP2
2230   D067 E1        ENDL2: POP HL
2240   D068 D1              POP  DE
2250   D069 C1              POP  BC
```

```
2260    D06A F1             POP  AF
2270    D06B C9             RET
2280                ;
2290                ;
2300                ;
2310    D06C 0D0A0A54 MENSG: DEFM 13,10,10,'TECLE [ESC] PARA ATIVAR',13,10
        D070 45434C45
        D074 205B4553
        D078 435D2050
        D07C 41524120
        D080 41544956
        D084 41520D0A
2320    D088 50524F47       DEFM 'PROGRAMA RESIDENTE...',7,13,10,10,0
        D08C 52414D41
        D090 20524553
        D094 4944454E
        D098 54452E2E
        D09C 2E070D0A
        D0A0 0A00
2330                ;
2340                ; PROGRAMA EXECUTOR
2350                ;
2360                ;
2370    D0A2 FE3A     EXCT:  CP   3AH            ; TESTA SE E [ESC]
2380    D0A4 C0              RET  NZ             ; VOLTA AO HOOK SE NAO
2390    D0A5 E5              PUSH HL
2400    D0A6 CD5601          CALL KILBUF         ; LIMPA BUFFER DE TECLADO
2410    D0A9 3EC9            LD   A,0C9H         ; CODIGO RET
2420    D0AB 32D1FD          LD   (HKEYA),A      ; COPIA P/ HOOK
2430    D0AE 3AFEFF          LD   A,(0FFFEH)
2440    D0B1 D3A8            OUT  (0A8H),A       ; CHAVEIA SLOT
2450    D0B3 CD0040          CALL 4000H          ; CHAMA PROG. RESIDENTE
2460    D0B6 3AFDFF          LD   A,(0FFFDH)
2470    D0B9 D3A8            OUT  (0A8H),A       ; CHAVEIA SLOT
2480    D0BB 3EC3            LD   A,0C3H         ; CODIGO CALL
2490    D0BD 32D1FD          LD   (HKEYA),A      ; COPIA P/ HOOK
2500    D0C0 E1              POP  HL
2510    D0C1 3E04            LD   A,4            ; RETORNA CARACT. SEM EFEITO
2520    D0C3 C9              RET
2530    D0C4          ENDX:  EQU  $
```

```
1000 ' REVISTA CPU - JUNHO 88
1010 ' PROGRAMAS RESIDENTES
1020 ' A.L. FREITAS
1030 '
1040 ' CODIGOS DO PROG. RESIDENTE
1050 '
1060 DATA C5,D5,E5,2A,DC,F3,E5,21
1070 DATA 00,00,11,00,50,01,BF,03
1080 DATA CD,59,00,CD,C3,00,21,04
1090 DATA 01,22,DC,F3,21,41,40,7E
1100 DATA FE,00,28,06,CD,A2,00,23
1110 DATA 18,F5,CD,9F,00,21,00,50
1120 DATA 11,00,00,01,BF,03,CD,5C
1130 DATA 00,E1,22,DC,F3,E1,D1,C1
1140 DATA C9,2A,2A,2A,2A,2A,2A,2A
1150 DATA 2A,2A,2A,2A,2A,2A,2A,2A
1160 DATA 2A,2A,2A,2A,2A,2A,2A,2A
1170 DATA 2A,2A,2A,2A,2A,2A,2A,2A
1180 DATA 2A,2A,2A,2A,2A,2A,2A,2A
1190 DATA 2A,00,0A,0A,09,09,52,45
1200 DATA 56,49,53,54,41,20,43,50
1210 DATA 55,00,0A,0A,09,09,4A,55
1220 DATA 4E,48,4F,20,38,38,00,0A
1230 DATA 0A,09,09,50,52,4F,47,52
```

```
1240  DATA 41,4D,41,53,20,52,45,53
1250  DATA 49,44,45,4E,54,45,53,0D
1260  DATA 0A,0A,09,09,41,2E,4C,2E
1270  DATA 20,46,52,45,49,54,41,53
1280  DATA 0D,0A,0A,0A,09,09,50,52
1290  DATA 45,53,53,49,4F,4E,45,20
1300  DATA 55,4D,41,20,54,45,43,4C
1310  DATA 41,0D,0A,0A,09,09,50,41
1320  DATA 52,41,20,53,41,49,52,2E
1330  DATA 0D,0A,0A,0A,2A,2A,2A,2A
1340  DATA 2A,2A,2A,2A,2A,2A,2A,2A
1350  DATA 2A,2A,2A,2A,2A,2A,2A,2A
1360  DATA 2A,2A,2A,2A,2A,2A,2A,2A
1370  DATA 2A,2A,2A,2A,2A,2A,2A,2A
1380  DATA 2A,2A,2A,2A,0D,0A,00
1390 '
1400 ' CODIGOS DO PROG. INSTALADOR
1410 '
1420  DATA F5,C5,D5,E5,CD,6C,00,21
1430  DATA A2,D0,11,CA,FF,01,22,00
1440  DATA ED,B0,DB,A8,32,FD,FF,21
1450  DATA 00,40,06,03,DB,A8,C6,04
1460  DATA 32,FE,FF,D3,A8,3E,AA,77
1470  DATA 56,2F,77,5E,7A,83,FE,FF
1480  DATA 28,0C,10,E8,3A,FD,FF,D3
1490  DATA A8,E1,D1,C1,F1,C9,21,00
1500  DATA C0,11,00,40,01,07,01,ED
1510  DATA B0,3A,FD,FF,D3,A8,21,CA
1520  DATA FF,22,D2,FD,3E,CD,32,D1
1530  DATA FD,21,6C,D0,7E,FE,00,28
1540  DATA 06,CD,A2,00,23,18,F5,E1
1550  DATA D1,C1,F1,C9,0D,0A,0A,54
1560  DATA 45,43,4C,45,20,5B,45,53
1570  DATA 43,5D,20,50,41,52,41,20
1580  DATA 41,54,49,56,41,52,0D,0A
1590  DATA 50,52,4F,47,52,41,4D,41
1600  DATA 20,52,45,53,49,44,45,4E
1610  DATA 54,45,2E,2E,2E,07,0D,0A
1620  DATA 0A,00,FE,3A,C0,E5,CD,56
1630  DATA 01,3E,C9,32,D1,FD,3A,FE
1640  DATA FF,D3,A8,CD,00,40,3A,FD
1650  DATA FF,D3,A8,3E,C3,32,D1,FD
1660  DATA E1,3E,04,C9
1670 ' CARREGADOR DE CODIGOS
1680 ' PARA MEMORIA
1690 '
1700 CLS: KEY OFF: WIDTH 40
1710 FOR I=&HC000 TO &HC106
1720 READ A$: A=VAL("&H"+A$)
1730 POKE I,A
1740 NEXT I
1750 FOR I=&HD000 TO &HD0C3
1760 READ A$: A=VAL("&H"+A$)
1770 POKE I,A
1780 NEXT I
1790 PRINT"PARA INSTALAR:": PRINT
1800 PRINT"DEF USR=&HD000": PRINT
1810 PRINT"A=USR(0)"
```

Estação de trabalho

JOSÉ AGUILERA

A popularização do uso dos computadores tem levado fabricantes e usuários a se preocuparem cada vez mais com os aspectos físicos, biológicos e psicológicos que, além das características técnicas do equipamento, influenciam o trabalho de computação.

O estudo das interrelações entre o corpo humano e os instrumentos de trabalho para aperfeiçoar o seu desenho é conhecido como Ergonomia (do grego ergon = trabalho). No nosso caso, esse estudo engloba equipamento, mobiliário e o próprio ambiente, visando criar condições que permitam eficiência sem fadiga.

As condições que se seguem pretendem dar uma rápida visão sobre o assunto e mostrar como certas recomendações próprias do trabalho profissional podem ser aplicadas ao uso do microcomputador pessoal para conseguir que longas sessões de digitação, processamento de textos ou mesmo de jogos, sejam realizadas confortavelmente e sem cansaço.

EQUIPAMENTO

É através do teclado e do monitor que nos comunicamos com o computador a maior parte do tempo e são os dois componentes que mais diretamente permitem o conforto ou desconforto do operador.

Existe equipamento que inclui num gabinete único a CPU, o monitor e o teclado, mas a apresentação mais popular nos microcomputadores parece ser a da CPU, monitor e teclado separados, ou CPU e teclado num gabinete e monitor separado.

TECLADO

A tendência dos fabricantes, inclusive dos aparelhos mais econômicos, é de aproximar o mais possível o formato dos teclados com o das máquinas de escrever elétricas, acrescentando teclas especiais, e, até, teclados numéricos. Aos poucos, foram abandonadas as tecladas tipo "chiclete" e, a não ser por razões especiais de manutenção e higiene, quase ninguém utiliza mais o teclado de membrana.

A disposição, forma, suavidade e textura das teclas deve produzir uma sensação agradável e de segurança ao operador. O "toque" das teclas, que, às vezes, pode tomar um caráter subjetivo, é um fator que deve ser levado em consideração, principalmente quando se trata de processamento de textos.

Como, geralmente, é nosso bolso quem orienta nossa compra, somente depois do uso do equipamento é que sentimos que alguma coisa poderia ter sido melhorada. Cabem, no entanto, algumas providências que podem facilitar o trabalho:

– Mantenha o click do teclado ligado. Isto aumenta a sua confiança.

– Situe o teclado um pouco afastado da beirada da mesa, a fim de deixar um espaço para descansar os pulsos.

– Caso sinta as teclas muito altas com relação à superfície da mesa, utilize um apoio para os pulsos, colocado na frente do teclado com uma altura de, mais ou menos, 2,5 cm.

MONITOR

É através da tela de vídeo que o computador se comunica conosco, e parece ter sido o motivo de queixa mais freqüente por parte dos operadores. Como a permanência prolongada em frente a um monitor constitui agressão para com os olhos, muitos estudos têm sido realizados para minimizar seus efeitos.

Os primeiros monitores apresentavam caracteres brancos sobre fundo preto ou inverso, mas após o uso mais ou menos prolongado, causavam irritação nos olhos. Os monitores de fósforo verde apareceram como uma solução e, até agora, são os mais populares, apesar de existirem pesquisas que parecem mostrar que a cor amarelo-âmbar seria a mais adequada.

Os monitores profissionais têm em geral, tela de vídeo de dez a quatorze polegares e possuem controles de contraste, brilho, vídeo invertido, quadro reduzido, etc., de modo que o operador possa regulá-los.

O uso de aparelhos de televisão como monitores se justifica como uma maneira de se iniciar a montagem de um equipamento pessoal, mas, quando se pretende realizar um trabalho mais sério, um monitor profissional é indispensável.

O uso de televisores adaptados ou com entrada de vídeo composto, significam um passo à frente. Neste caso, televisores coloridos são convenientes para a utilização com editores gráficos ou com programas nos quais as cores são fundamentais. Os monitores coloridos RGB ainda são muito caros para um esquema doméstico.

Finalmente, o uso de filtros da cor verde ou âmbar em televisores branco e preto pode se converter em arma de dois

gumes. Com a limpeza e o manuseio, o plástico acaba perdendo a transparência e produzindo imagens duplas ou fora de foco, que se tornam motivo de fadiga.

MOBILIÁRIO

A mesma preocupação que se tem para com o equipamento deve existir com relação ao mobiliário que será utilizado no trabalho com o computador.

Existe uma grande semelhança entre o ato dedigitar e o de bater à máquina. Por isto, de forma geral, móveis confortáveis para datilografia são confortáveis para computação.

CADEIRA

De pouco adianta o investimento feito com o equipamento se, ao cabo de uma hora de trabalho, estamos com as costas ou os ombros doloridos e sem vontade de continuar em conseqüência do uso de uma cadeira inadequada. Uma cadeira para computador deve ter assento e encosto estofados, altura regulável, encosto baixo, de preferência também regulável. Deve ser giratória e apoiada sobre rodízios. A base de cinco pés é mais estável que a base de quatro.

MESA

Em se tratando de um equipamento que integre num gabinete único a CPU, o teclado e o monitor, ou que permita a colocação do monitor em cima da CPU, uma mesa de datilografia com suficiente espaço pode ser o bastante. O importante é que o teclado deve ficar na altura usada para a máquina de escrever e o monitor ligeiramente mais alto. Existem mesas apropriadas que possuem suportes com diferentes alturas, fixos ou reguláveis.

Acontece com freqüência que o microcomputador é instalado num pequeno espaço que se conseguiu arrumar em casa ou no escritório. Neste caso, é necessária a adaptação às condições existentes, levando em consideração algumas recomendações:

O teclado deve ficar na altura ou ligeiramente abaixo dos cotovelos. A mesa com 65 a 68 cm. (Um teclado colocado sobre uma escrivaninha comum pode se tornar um instrumento de tortura).

– É conveniente que a mesa para o teclado seja suficientemente grande para permitir o descanso dos pulsos e deixar espaço para o uso do mouse. Utilizar o mouse sobre uma escrivaninha comum torna-se cansativo porque a altura da mesa dificulta o movimento livre do braço.

– A distância entre os olhos e o monitor deve ficar entre 45 e 65 cm.

AMBIENTE DE TRABALHO

As condições de iluminação, temperatura, nível de barulho e a própria decoração do ambiente de trabalho são fatores que influenciam conforto dos operadores e sua eficiência.

O local de trabalho deverá ser afastado do ruído do trânsito, pintado com cores claras e neutras e, em se tratando de instalações profissionais, deverá ter controle de temperatura e umidade.

Quanto à iluminação, são três os tipos de luz que devem ser levados em consideração: luz ambiental, luz da área de trabalho e a luz proveniente da tela de vídeo do monitor. A luz ambiental deverá ser suave de modo a permitir que o operador descanse os olhos desviando a vista do monitor ou da área de trabalho, que deverá ser mais iluminada. Uma arrumação cuidadosa deverá ser feita para evitar reflexos das fontes de luz, de janelas ou objetos brilhantes na tela.

Das condições de iluminação dependerá o grau de fadiga dos olhos do operador.

CONSIDERAÇÕES FINAIS

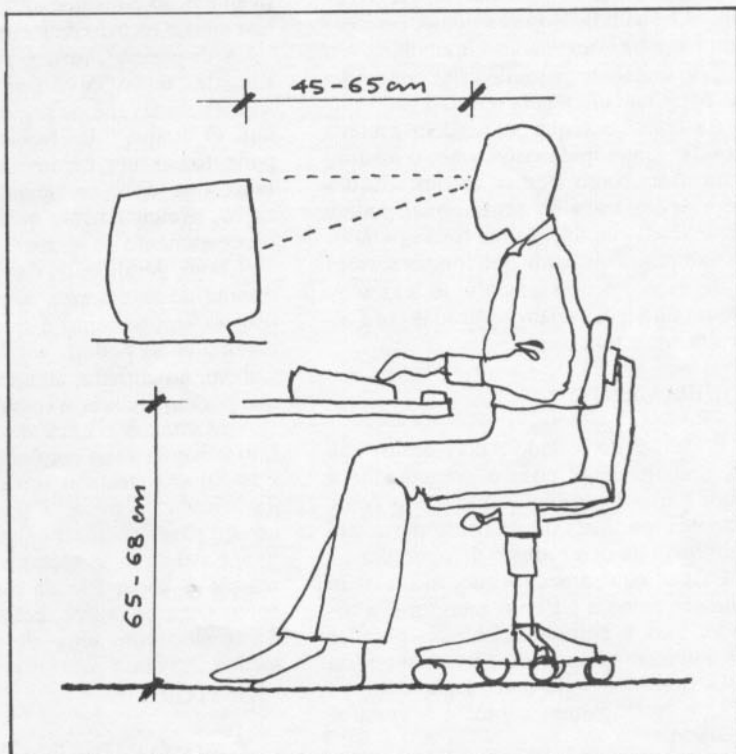
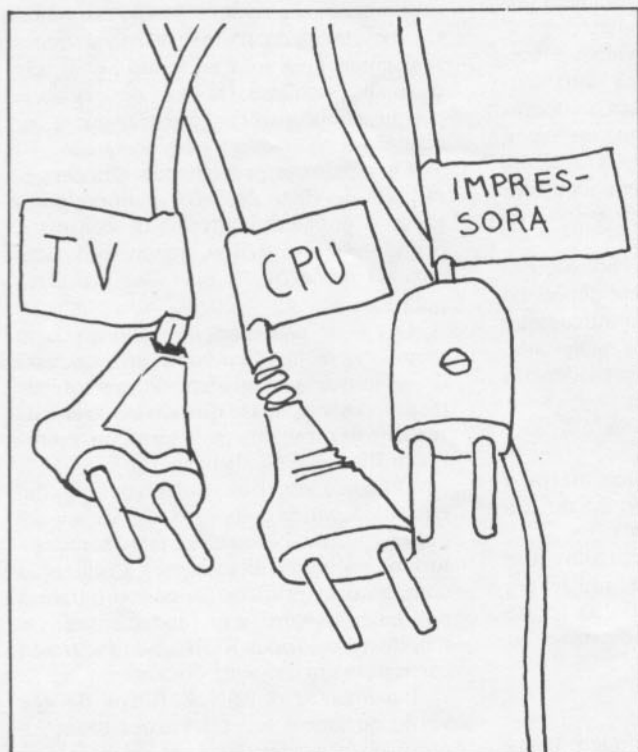
A instalação de uma estação de trabalho com finalidade profissional exige o cumprimento de normas e recomendações nem sempre aplicáveis quando se trata da adaptação de um pequeno espaço em casa ou no escritório. Vale a pena, no entanto, levantar alguns pontos que podem melhorar a sua eficiência e satisfação no uso do microcomputador:

– Aplique, no possível, as orientações dedicadas ao campo profissional.

– Use uma cadeira de datilografia mesmo para trabalhos esporádicos.

– Use um monitor de fósforo verde ou âmbar para trabalhos mais sérios.

– Mantenha em ordem a sua estação de trabalho. Coloque etiquetas de identificação nos cabos e fios. Enrole e prenda o excedente dos fios.

– Lembre-se que o equipamento determina as suas possibilidades técnicas, mas a sua eficiência dependerá das suas condições de trabalho.

A trilha Ø

GONÇALO R. F. MURTEIRA

Com o início da comercialização dos drives para a linha MSX, um novo horizonte foi aberto para os usuários desta linha de equipamentos.

A operação com disco não oferece dificuldade de tipo algum, possuindo comandos simples e de fácil assimilação.

Podemos dizer que a trilha 0 é alma do disquete e, por esta razão, resolvemos fornecer maiores informações sobre esta parte vital do disco para que você tenha maiores condições para resolver os problemas que lhe venham a aparecer, tal como recuperar um arquivo deletado acidentalmente.

	Face simples	Face dupla
Nº de trilhas	40	40
Número de setores por trilha	9	9
Número de setores por disquete	360	720
Número de setores disponíveis	351	708
Número de setores reservados	9	12
Número de setores do boot	1	1
Número de setores do FAAT	4	4
Número de setores do diretório	4	7

Após formatarmos um disco no MSX DOS, o mesmo será dividido conforme ilustra a figura

Às vezes, ao formatarmos um disco, pode acontecer que não obtenhamos a capacidade total de armazenamento. Quando isto ocorre, é porque o computador, ao efetuar a verificação da formatação, encontrou alguma trilha não apta para uso, eliminando-a para que não se venha a ter surpresas no futuro.

Pode ocorrer que o dano no disco ocorra na trilha 0, fazendo com que não se possa utilizar mais aquele disco. Um exemplo prático desta situação é quando tentamos obter o diretório de um disco formatado e conseguimos a mensagem de erro de entrada e saída.

O DIRETÓRIO

O diretório está alocado na trilha 0, nos setores de 5 a 8, caso o disco esteja formatado em face simples, ou nos setores de 5 a 8 do lado 0 e de 0 a 2 do lado 1, caso a formatação do disco seja de face dupla.

Como já é conhecido, os micros da linha MSX e IBM PC são compatíveis a nível de organização de disco. Devido a esta particularidade, algumas informações disponíveis no diretório do MSX não são utilizadas, como é o caso do atributo e hora de gravação.

Ao formatarmos um disco, definimos, também, o espaço fixo reservado para o diretório. Este espaço é de 7 setores para os discos de face dupla ou de 4 setores para formatação em face simples.

Cada setor tem capacidade para armazenar 512 bytes. Portanto, o diretório pode armazenar até 112 nomes de arquivos em discos de face dupla ou 64 em discos de face simples.

Assim sendo, se você possuir 112 arquivos gravados em um disco de face dupla, mesmo que ainda exista espaço disponível, você não conseguirá gravar o 113º programa, pois não haverá mais espaço disponível no diretório para o armazenamento das informações referentes aquele arquivo. Neste caso, você será informado pelo sistema operacional através da mensagem diretório cheio.

PROTEÇÃO DE PROGRAMAS

No IBM-PC temos a facilidade de poder proteger um programa contra os olhares dos curiosos e, assim, tentar evitar a cópia.

Para fazermos o mesmo no MSX, basta mudarmos o byte referente ao tributo para qualquer valor entre 04H e 07H.

Com este procedimento, o programa poderá ser lido, mas não aparecerá se for solicitado o diretório do disco.

RECUPERAÇÃO DE PROGRAMAS

Pode acontecer que, acidentalmente, você apague um arquivo e deseje recuperá-lo. Se, após apagar um arquivo, você não tiver efetuado nenhuma gravação no disco, poderá recuperá-lo, pois o comando que apaga um arquivo simplesmente troca o primeiro caracter do nome por E5H.

Para recuperarmos o arquivo basta trocarmos novamente o primeiro caracter pelo código original.

Sobre a trilha 0 de um disco ainda existe muito para se falar e ela será tema de outros artigos de edições futuras.

Impressão Dupla

J. L. FONSECA

Você já deve ter tido a necessidade de efetuar o reforço de um texto impresso.

A rotina que apresentamos foi baseada em um programa publicado no livro "Cem dicas para MSX", na página 120, da Editora Alpeh, no qual efetuamos alterações.

O programa permite que o usuário selecione o número de colunas e o número de vezes que a cabeça de impressão deverá efetuar um reforço do texto, imprimindo novamente os pontos já impressos.

Portanto, com este programa, você pode fazer com que as suas listagens ou textos sejam impressos com mais qualidade e contraste.

O efeito obtido é diferente de uma impressão em qualidade de carta, na qual cada ponto é impresso duas vezes, mas havendo um ligeiro espaçamento entre cada ponto.

```
115 SCREEN 0:CLEAR 200,&HE000
116 PRINT:PRINT:PRINT
117 PRINT"CARACTERES POR LINHA ";
118 INPUT C
119 IF C<1 OR C>255 THEN GOTO 1160
120 PRINT"NUMERO DE PASSAGENS ";
121 INPUT P
122 IF P<1 OR P>255 THEN GOTO 1200
123 FOR K=&HE000 TO &HE073
124 READ A$:POKE K,VAL("&H"+A$)
125 NEXT K
126 POKE &HE022,C
127 POKE &HE045,P
128 DEFUSR0=&HE066
129 I=USR0(0)
1000 DATA F5,C5,D5,E5,DD,E5,DD,21
1010 DATA 65,E0,FE,0D,28,4D,FE,0A
1020 DATA 28,13,06,00,DD,4E,00,21
1030 DATA 66,E0,09,03,DD,71,00,77
1040 DATA 79,FE,00,20,36,3E,C9,32
1050 DATA B6,FF,0E,00,DD,46,00,78
1060 DATA B0,28,15,21,66,E0,7E,CD
1070 DATA A5,00,23,10,F9,3E,0D,CD
1080 DATA A5,00,0C,79,FE,00,20,E4
1090 DATA 3A,65,E0,B7,3E,0A,C4,A5
1100 DATA 00,0E,00,DD,71,00,3E,C3
1110 DATA 32,B6,FF,DD,E1,E1,D1,C1
1120 DATA F1,33,33,B7,C9,00,F3,21
1130 DATA 00,E0,22,B7,FF,3E,C3,32
1140 DATA B6,FF,FB,C9,00,FF,00,FF
```

Programação gráfica em SCREEN 1

SÍLVIO CHAN

Que tal fazer gráficos instantâneos em uma tela de texto?

A princípio, você deve estar pensando que estou propondo algo impossível. Não nego que você esteja parcialmente correto. No entanto, o que este artigo pretende ensinar, não é nada menos do que uma das técnicas mais utilizadas na confecção de telas e cenários gráficos nos jogos em Assembler, os quais você certamente deve ter em casa.

A principal vantagem desta técnica é a rapidez com que as telas são desenhadas. Esta técnica se baseia na criação de blocos gráficos que podem ser coloridos com as cores do MSX. No nosso caso, a criação destes blocos será feita em SCREEN 1, que é uma tela de texto do MSX.

Para compreender o modo como são criados os blocos, é preciso entender o funcionamento das tabelas de Formas dos Caracteres e de cores do SCREEN 1, já que os blocos serão formados através da alteração do desenho de cada caracter. Isto significa que podemos criar, nada menos, do que 255 tipos diferentes de blocos.

O desenho de cada caracter é armazenado na tabela de forma em grupos de oito bytes. Sua extensão é de 2 Kb.

A tabela de cores armazena em um byte a cor de um grupo de oito caracteres.

Como no MSX existem 256 caracteres, esta tabela possui a extensão de 32 bytes. O valor contido em cada um destes bytes é calculado pela seguinte fórmula:

$$V = 16 * CE + CF,$$

onde CE é a cor de frente e CF é a cor de fundo.

Para livrar os leitores do trabalho de calcular os endereços e respectivos valores, este artigo é acompanhado de um programa muito eficiente para a criação dos blocos gráficos.

Após digitá-lo, salve-o e execute.

O PROGRAMA

Ao rodar o programa o micro perguntar-lhe-á se você deseja carregar um banco de blocos ou se deseja ir direto para o editor. Caso já exista um banco de blocos na memória, também surgirá a opção para poder utilizá-lo. Em seguida, aparecerá um menu. Para efetuar a escolha da opção desejada, você deverá utilizar a seta para baixo e posicionar a barra de espaço na opção desejada.

Depois de ter editado o seu banco, salve-o. Para recomeçar o programa, pressione F1 e, para sair, F2.

Para utilizar um banco já gravado, digite BLOAD "nome do banco", r. Após o carregamento, os caracteres serão automaticamente modificados, devido a uma rotina em Assembler existente no final do programa que faz exatamente isto.

Com o banco já carregado na memória, basta usar a instrução PRINT para imprimir os blocos desejados (caracteres) na tela.

Caso você mude acidentalmente de tela, basta digitar:

DEFUSR = &hE813:A = USR (0) ou
DEEUSR = &hD813:A = USR (0)

para quem estiver utilizando disco.

Programas criados para manipular o banco de caracteres, como por exemplo jogos, aplicativos e adventures, devem ter suas extensões calculadas para não apagarem o banco.

Os usuários de disco devem efetuar as seguintes alterações:

Linha 90 – alterar o valor das variáveis E1, E2, E3 e E4 para &hDOOO, &hD82E, &hD813 e &hD7F1, respectivamente.

Linha 150 – troque &hEO por &hOO e &hE7 por &hD7.

```
20 ' Gerador de Caracteres
30 '
40 ' Silvio Chan -- 1988
60 '
70 ' INICIALIZA - E1 - END. INICIAL - E
2- END. FINAL - E3- END. EXEC. - E4 - E
ND. TAB. CORES
80 '
90 CLEAR:KEYOFF:COLOR7,1,1:SCREEN1,2,1:
WIDTH30:E1=&HE000:E2=&HE82E:E3=&HE813:E
4=&HE7F1:DEFUSR=E3
100 LOCATE0,10:IFPEEK(E3)=&HCDTHENPRINT
"  <m> Caracteres na Mem"ria    <l> Lo
ad      <i> Inicia"ELSEPRINT"  <l> Loa
d      <i> Inicia"
110 GOSUB1190:IFA$="l"ORA$="L"THENCLS:B
LOAD"cas:",R:GOTO160
120 IFA$="i"ORA$="I"THENCLS:GOTO160
130 IF(PEEK(E3)=&HCD)ANDA$="m"ORA$="M"T
HENA=USR(0):GOTO160
140 GOTO110
150 DATA CD,6F,0,11,0,0,21,0,E0,1,F0,7,
CD,5C,0,11,0,20,21,F1,E7,1,20,0,CD,5C,0
,C9
160 FORI=E3TOE2:READA$:POKEI,VAL("&H"+A
$):NEXT
170 LOCATE0,10:PRINT"PROGRAMA GERADOR D
E CARACTERES"
180 '
190 ' CRIA SPRITES REPRESENTATIVOS
200 '
210 FORT=0TO3:A$="":B$="":FORI=0TO15:RE
ADA:A$=A$+CHR$(A):NEXTI:FORI=0TO15:READ
B:B$=B$+CHR$(B):NEXTI:SPRITE$(T)=A$+B$:
NEXT
220 DATA 255,128,128,128,140,140,128,12
8,143,144,160,255,249,241,249,255
```

```
230 DATA 255,1,1,1,49,49,1,1,241,9,5,25
5,159,143,159,255
240 DATA 255,252,243,239,216,192,192,19
2,216,216,221,222,223,231,248,255
250 DATA 255,63,207,247,27,3,3,3,27,27,
155,219,251,231,31,255
260 DATA 255,128,191,190,188,188,185,18
5,176,176,179,179,179,191,128,255
270 DATA 255,1,253,125,61,61,157,157,13
,13,205,205,205,253,1,255
280 DATA 255,254,252,249,243,243,243,25
5,255,255,255,255,255,255,255,255
290 DATA 255,63,31,207,231,231,231,199,
143,31,63,63,255,63,63,255
300 ONKEYGOSUB1250,1150:KEY(1)ON:KEY(2)
ON
310 SPRITE$(4)=STRING$(32,255)
320 '
330 ' MOSTRA MENU PRINCIPAL
340 '
350 GOSUB1090:FORI=1TO4:PUTSPRITEI-1,(4
0,I*40),15,I-1:NEXT:PUTSPRITE4,(40,40),
12,4
360 LOCATE7,6:PRINT"Salvar caracteres":
LOCATE7,11:PRINT"Colorir o caractere"
370 LOCATE7,16:PRINT"Redefinir o caract
ere":LOCATE7,21:PRINT"Codigo dos caract
eres":X=40
380 '
390 ' SELECIONA UMA OPCAO
400 '
410 IFSTRIG(0)=-1THEN450
420 IFSTICK(0)=5ANDX<160THENX=X+40ELSEI
FX=160ANDSTICK(0)=5THENX=40ELSE410
430 PUTSPRITE4,(40,X),12,4:FORI=0TO150:
NEXT
440 GOTO420
450 X=X/40
460 ONXGOSUB510,640,820,980
470 GOTO350
480 '
490 ' GRAVOO DOS CARACTERES
500 '
510 GOSUB1100:GOSUB1090
520 IFINKEY$<>""THEN520
530 GOSUB1090:INPUT"Nome do programa (6
)";N$
540 IFN$=""THEN530
550 PRINT:PRINT"      Espere um instante
560 FORI=0TO2032:POKEE1+I,VPEEK(I):NEXT
570 FORI=8192TO8223:A=E4+(I-8192):POKEA
,VPEEK(I):NEXT
580 PRINT"                  ok!!"
590 BSAVEN$,E1,E2,E3
600 GOTO1190
610 '
620 ' COLORIR CARACTERES
630 '
640 GOSUB1100:GOSUB1090:N=0:V=8192
650 IFINKEY$<>""THEN650
660 PRINT"Verifique o conjunto de carac
-teres a ser  colorido  atravesdas seta
s.  "
670 IFSTRIG(0)=-1THEN710
680 IFSTICK(0)=3ANDN<246THENN=N+8:V=V+1
ELSEIFSTICK(0)=3ANDN>=246THENN=0:V=8192
690 IFSTICK(0)=7ANDN>0THENN=N-8:V=V-1EL
SEIFSTICK(0)=7ANDN=0THENN=247:V=8223
700 FORI=0TO7:VPOKEBASE(5)+289+I,N+I:N
EXT:GOTO670
710 IFINKEY$<>""THEN710
720 LOCATE0,14:INPUT"Cor de frente (0-1
5) ";CA
730 LOCATE0,16:INPUT"Cor de fundo (0-15
) ";CF
740 IFCA<0ORCA>15ORCF<0ORCF>15THEN640
750 Q=16*CA+CF:VPOKEV,Q
760 LOCATE0,21:PRINT"  Continua a color
ir (s/n)"
770 IFINKEY$<>""THEN770
780 A$=INKEY$:IFA$="S"ORA$="s"THEN640EL
SEIFA$="N"ORA$="n"THENRETURNELSE780
790 '
800 ' REDEFINE CARACTERE
810 '
820 GOSUB1100:GOSUB1090
830 IFINKEY$<>""THEN830
840 INPUT"Codigo ASCII (0-254) ";C
850 IFC>254ORC<0THEN820
860 V=C*8:FORJ=0TO7
870 A$(J)=BIN$(VPEEK(J+V)):B=LEN(A$(J))
880 IFB<8THENLETA$(J)=LEFT$("00000000",
8-LEN(A$(J)))+A$(J)
890 VPOKEBASE(5)+161,C:NEXTJ:LOCATE0,7
900 FORJ=0TO7:PRINTA$(J),"00000000":NEX
TJ
910 FORI=0TO7:LOCATE13,I+7:LINEINPUTA$
920 P$=STR$(VAL("&B"+RIGHT$(A$,8))):VPO
KEI+V,VAL(P$):NEXT
930 LOCATE0,20:PRINT"  Continua a redef
inir (s/n)"
940 A$=INKEY$:IFA$="s"ORA$="S"THEN820EL
SEIFA$="n"ORA$="N"THENRETURNELSE940
950 '
960 ' CONSULTA TABELA ASCII
970 '
980 GOSUB1100:GOSUB1090
990 FORI=0TO32:PRINTI;") ";CHR$(1)+CHR$
(I+64),:NEXT
1000 IFINKEY$<>""THEN1000
1010 A$=INKEY$:IFA$<>CHR$(32)THEN1010EL
SEN=33
1020 GOSUB1090:FORI=NTON+31:PRINTI;") "
;CHR$(I),:NEXT
1030 IFINKEY$<>""THEN1030
1040 A$=INKEY$:IFA$<>CHR$(32)THEN1040EL
SEN=N+32
1050 IFN=225THEN1060ELSE1020
1060 GOSUB1090:FORI=225TO254:PRINTI;")
";CHR$(I),:NEXT
1070 A$=INKEY$:IFA$<>CHR$(32)THEN1070EL
SERETURN
1080 GOTO1080
1090 CLS:PRINT"Gerador/MSX  - Revista C
PU 02 ";STRING$(30,195):RETURN
1100 FORI=0TO4:PUTSPRITEI,(-32,-32):NEX
T:RETURN
1110 GOSUB1190:GOTO510
1120 '
1130 ' FINALIZA PROGRAMA - KEY 2
1140 '
1150 SCREEN0:END
1160 '
1170 ' ROTINA DE LEITURA DO TECLADO
1180 '
1190 IFINKEY$<>""THEN1190
1200 LOCATE8,20:PRINT"Pressione Tecla"
1210 A$=INKEY$:IFA$=""THEN1210ELSERETUR
N
1220 '
1230 ' REEXECUTA O PROGRAMA - KEY 3
1240 '
1250 SCREEN1:RUN
```

MÁXIMAS E MÍNIMAS

J. L. FONSECA

Na coluna de hoje, vamos falar sobre um problema que muitas vezes nos aparece quando o nosso programa exige uma entrada de dados pelo teclado.

Muitas vezes o usuário digita algo errado e o programa, ao tentar usar o dado errado, pára com uma mensagem de erro.

Ora, um programa não deve parar devido a erros na entrada, devendo, em vez disso, fazer uma crítica da entrada e dar ao usuário a oportunidade de dar a entrada correta.

Temos dois tipos distintos de entradas com que nos preocuparmos: as numéricas e as alfanuméricas. No primeiro caso, devemos verificar se o valor fornecido está dentro da faixa permitida para o mesmo no programa. No segundo caso, devemos verificar o comprimento da string.

Em qualquer dos casos é uma boa idéia apresentar na tela o campo onde se dará a entrada com algum tipo de caracter, como o ponto, por exemplo, para indicar o número de posições possíveis, não devendo ser possível ao usuário sair do campo.

Este tipo de preocupação pode parecer excesso de cuidado, mas é, na verdade, um modo simpático de guiar a pessoa que está usando o seu programa pela primeira vez, além de tornar mais difícil que uma entrada errada se infiltre e pare o seu programa com uma mensagem de erro, que, em geral, não explica muito sobre a causa.

No MSX temos vários comandos e funções que nos ajudam nesta tarefa. Para a entrada de dados, temos a função INKEYS que nos permite fazer a crítica de cada caracter à medida que o mesmo é digitado. Podemos, por exemplo, não aceitar a digitação de letras num campo numérico ou não permitir que um campo alfanumérico tenha mais que o comprimento permitido.

Muitos leitores já devem estar reclamando que a função INKEY$ só permite a entrada para variáveis alfanuméricas e, assim mesmo, só de um caracter por vez. No entanto, esta aparente desvantagem é a sua principal virtude, pois, assim, nos permite editar a entrada e, nos casos de valores numéricos testar a sua validade ao nível de cada dígito sem o perigo de erros de formato ou de valor.

Como exemplo, damos, a seguir duas listagens de rotinas de entrada de dados: uma para números inteiros e outra para cadeias de caracteres.

Estas rotinas não são as melhores ou mais eficientes, mas servem para dar uma idéia do que é possível fazer, indicando o caminho para quem quiser usá-las na sua forma original ou modificadas de modo a se adaptarem melhor à aplicação em uso.

```
10 ' REVISTA CPU
20 ' MAXIMAS E MINIMAS
30 ' J.L. FONSECA
40 ' ROTINAS 1 E 2
50 '
90 CLS
1000 ' ROTINA PARA ENTRADA DE STRINGS
     COM EDICAO
1010 ' CO,LI COLUNA E LINHA EM QUE SE
     QUER A ENTRADA
1020 ' IN COMPRIMENTO MAXIMO DA
     STRING
1030 ' IS$ STRING ONDE ESTA A SAIDA
1040 LOCATE CO,LI : FOR II=1 TO IN :
    PRINT ".";: NEXT II : IS$=""
1050 LOCATE CO,LI : IP=1
1060 II$=INKEY$ : IF II$="" THEN GOTO
    1060
1070 IF II$=CHR$(8) OR II$=CHR$(127)
    THEN GOSUB 3030 : GOTO 1060
1080 IF II$=CHR$(13) OR IP>IN THEN
    RETURN
1090 IF II$<CHR$(32) OR II$>CHR$(127)
    THEN GOTO 1060
1100 PRINT II$; : IS$=IS$+II$ : IP=IP+1
    : GOTO 1060
2000 ' ROTINA PARA ENTRADA DE NUMEROS
     INTEIROS
2010 ' TUDO COMO NA ROTINA ANTERIOR
     EXCETO QUE A SAIDA SE DA POR II%
2020 ' MA,MI VALORES MAXIMOS E MINIMOS
     QUE A ENTRADA PODE TER
2030 IN=6
2040 LOCATE CO,LI : FOR II=1 TO IN :
    PRINT "."; : NEXT II : IS$=""
2050 LOCATE CO,LI : IP=1
2060 II$=INKEY$ : IF II$="" THEN GOTO
    2060
2070 IF II$=CHR$(8) OR II$=CHR$(127)
    THEN GOSUB 3030 : GOTO 2060
2080 IF II$=CHR$(13) OR IP>IN THEN 2110
2085 IF II$="-" AND IP=1 THEN GOTO 2100
2090 IF II$<CHR$(48) OR II$>CHR$(57)
    THEN GOTO 2060
2100 PRINT II$; : IS$=IS$+II$ :
    IP=IP+1 : GOTO 2060
2110 II=VAL(IS$) : IF II>=MI AND II<=MA
    THEN RETURN
2120 LOCATE CO,LI : IF II<MI THEN PRINT
    "BAIXO"
2130 IF II>MA THEN PRINT "ALTO"
2140 FOR II=1 TO 256 : NEXT II : LOCATE
    CO,LI : GOTO 2000
3000 '
3010 ' ROTINA PARA TRATAR DO DELETE
3020 '
3030 IF IP=1 THEN RETURN
3040 IP=IP-1 : LOCATE IP+CO-1,LI :PRINT
    "." : LOCATE IP+CO-1,LI
3050 IS$=LEFT$(IS$,IP-1):RETURN
```

Múltipla Escolha

CARLOS E. A. MOREIRA

A idéia desta seção surgiu com o aparecimento de compiladores de outras linguagens para o MSX. Já encontramos com facilidade pelo menos um compilador de linguagem C e um compilador Pascal para o MSX.

Aqui, mostraremos um método de ordenação que, apesar de não ser muito rápido, é bem simples e funcional para a maioria dos casos. O método é o "bubble sort", ou seja, ordenação tipo bolha, tendo surgido o nome da comparação do método em que os valores a serem ordenados sobem a lista de valores com o fato de uma bolha subir à superfície da água (estranho, não é mesmo?). A rotina de ordenação é apresentada em Basic, Pascal e C, não tendo esta seção o objetivo de ensinar qualquer uma destas linguagens, mas, somente, apresentar rotinas para o usuário selecionar aquela que mais se enquadrar ao seu conhecimento. Bom proveito!

O algoritmo é o seguinte: a lista é pesquisada do início ao final. Cada vez que um valor mais alto da lista se encontra numa posição baixa, este é deslocado, trocado com o seguinte. O método se repete até que toda a lista esteja ordenada.

BUBBLE SORT EM BASIC

```
1000 REM BUBBLE SORT
1005 REM MSX SYSTEMS
1010 REM DEVEM SER PASSADAS
AS SEGUINTES
1015 REM VARIÁVEIS A ESTA RO-
TINA:
1020 REM ITEM - VETOR A ORDE-
NAR
1025 REM CONT - NO. DE ELEMEN-
TOS
1030 REM
1040 FOR A%=1 TO COUNT-1
1050 FOR B%=COUNT-1 TO A
STEP-1
1060 IF ITEM (B%) > ITEM (B%+1)
THEN SWAP ITEM (B%), ITEM
B%+1)
1070 NEXT B,A
```

BUBBLE SORT EM PASCAL

```
{ na definição de variáveis devem estar
as linhas abaixo }
var
   a, b, temp, conta: integer;
   item: array [1..XX] of TIPO
{ XX é o nº de elementos do vetor e
TIPO é o tipo da variável: inteiro, real,
etc. }
{ a rotina ordenadora se encontra abai-
xo }
{ conta é a variável que armazena o nº
de elementos do vetor }
for a:=1 to conta-1 do
   for b:=conta-1 downto a do
      if (item[b] > item [b+1]) then
         begin
            temp:=item [b];
            item [b]:=item [b+1];
            item [b+1]:=temp;
         end;
```

BUBBLE SORT EM C

Seguindo a estrutura da linguagem C, apresento uma função para a ordenação.

A variável item é um ponteiro para um vetor interno. Count é uma variável inteira contando o número de elementos do vetor.

Estes valores devem ser passados para a função na sua chamada /*.

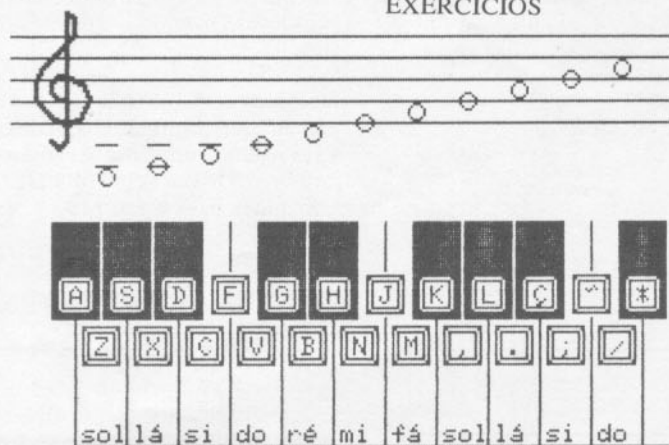
```
Void bubble (item, count)
int count,
   *item;

{
int a,
   b,
   t;
for (a=1; a < count; + + a) {
   for (b=count-1; b > item [b])
      { t = item [b-1];
      item [b-1] = item [b];
      item [b] = t;
      {
   {
{
```

LIVROS

CURSO DE MÚSICA MSX – Teoria e Prática
Editora ALEPH – 1988 – Barbieri e Piazzi

O "CURSO DE MÚSICA MSX" é um livro escrito para o usuário que quer aprender música usando como instrumento musical um micro MSX.

Além do caráter marcadamente didático da obra, chama a atenção do leitor o excelente trabalho de ilustração. O livro é fartamente ilustrado e cada tópico de teoria é acompanhado de exemplos e exercícios práticos. Abaixo, podemos observar a ilustração da página 97 do livro, gerada por um programa que transforma o teclado do micro num piano.

A grande vantagem do leitor ao usar um MSX como instrumento musical é a eliminação do enfadonho treinamento psico-motor necessário para o domínio de instrumentos tradicionais. Enquanto que para tocar piano, violão ou outro instrumento qualquer são necessárias horas de treino, "tocar" um MSX é algo muito mais fácil, estando ao alcance de qualquer pessoa que tenha um conhecimento mínimo de BASIC. O CURSO DE MÚSICA parte do pressuposto que o leitor saiba apenas usar o micro, mesmo que de forma bastante precária. A apresentação dos programas listados é feita de forma a tornar o mais fácil possível a digitação, sendo explicados detalhadamente. O comando básico para a geração de tons é sobejamente detalhado num apêndice no próprio livro e, como os autores advertem, mesmo os leitores que não sabem BASIC ao iniciar a leitura, acabam aprendendo ao longo do livro, ainda que de forma rudimentar.

Se você quer aprender música, este é mais um lançamento que não pode passar em branco!

A seguir, comentamos, rapidamente, o conteúdo de cada capítulo.

AULA 1 – A MÚSICA E O MSX

Quase sempre, ao longo do livro, a abordagem da teoria segue um caminho histórico. Neste capítulo, os autores introduzem o leitor num universo da música. Os subitens são os seguintes:

INTRODUÇÃO HISTÓRICA
ALGUNS CONCEITOS BÁSICOS
OS INSTRUMENTOS MUSICAIS
OS SONS DO MSX
EXERCÍCIOS

AULA 2 – A NOTAÇÃO MUSICAL PARA A ALTURA

Neste capítulo são abordadas as propriedades características dos sons e algumas notações usadas para representá-las. Os subitens são os seguintes:

INTRODUÇÃO
PROPRIEDADES DO SOM
A NOTAÇÃO MUSICAL
A NOTAÇÃO PARA A ALTURA
A EXTENSÃO DOS INSTRUMENTOS MUSICAIS
A EXTENSÃO DA VOZ HUMANA
AS NOTAÇÕES ALEMÃ E INGLESA
EXERCÍCIOS

AULA 3 – A NOTAÇÃO MÚSICAL PARA A DURAÇÃO

Neste capítulo, mais algumas notações são introduzidas e a representação das notações nos micros MSX é comentada.

AS FIGURAS
PAUSAS
ALGUMAS CONVENÇÕES
AS NOTAÇÕES PARA A INTENSIDADE E TIMBRE
AS REPRESENTAÇÕES NO MSX
EXERCÍCIOS

AULA 4 – A DIVISÃO DA MÚSICA

Este capítulo aborda a divisão da música em ciclos repetitivos e introduz os conceitos de compasso, andamento e ritmo.

TEMPOS E COMPASSOS
ANDAMENTO
VAMOS CONSTRUIR UM METRÔNOMO
RITMO
SINAIS DE REPETIÇÃO
EXERCÍCIOS

AULA 5 – AS ESCALAS MUSICAIS

Aqui os autores apresentam o surgimento e desenvolvimento das escalas musicais que culminaram com a atual escala musical temperada.

MODOS
MÚSICA, MATEMÁTICA E FILOSOFIA
AS TECLAS PRETAS DO PIANO
TOM E SEMITOM
EXERCÍCIOS

AULA 6 – OS ACIDENTES MUSICAIS

Este capítulo aborda as notações usadas para indicar mudanças mais específicas ou mais sutis durante a execução de uma música. Os termos mais usuais e algumas notações mais modernas são comentados.

OS ACIDENTES
NOVAS ESCALAS
ALGUMAS CURIOSIDADES
O MSX "BEM TEMPERADO"
EXERCÍCIOS

APÊNDICE A – EXERCÍCIOS SUPLEMENTARES
APÊNDICE B – RESPOSTAS DOS EXERCÍCIOS
APÊNDICE C
1) O COMANDO PLAY
2) A FUNÇÃO PLAY

O Livro Vermelho do MSX – "THE RED BOOK"
Editora McGRAW-HILL
Avalon Software
17 x 24 cm – 323 páginas

Lançado em 1985 pela Avalon Software, chega agora, ao Brasil, através da editora McGraw-Hill, o famoso livro vermelho do MSX, a Bíblia do MSX, onde são colocadas à disposição dos leitores informações totalmente detalhadas do software e hardware do MSX, sendo dirigido aos usuários com nível de programas avançado que procuram formas para resolver problemas cuja solução não é encontrada em manuais e que, a princípio, podem parecer insolúveis.

Da maneira como são apresentadas as informações, o leitor assimila facilmente a idéia apresentada e termos como PPI, VDP, BIOS, porta de entrada e saída..., passarão a ser tão familiares como a instrução PRINT.

Os três primeiros capítulos dizem respeito à interface periférica programável, processador de display de vídeo e o gerador de som programável, que constituem a interface entre o Z80 e o hardware periférico. Nos demais capítulos temos a descrição da BIOS (Basic Input/Output System – sistema básico de entrada e saída) do interpretador Basic MSX da Microsoft e alguns exemplos de programas em código de máquina que utilizam recursos da ROM.

No capítulo referente à BIOS em ROM é apresentada uma descrição funcional de cada rotina em separado. As rotinas "padrões" receberão uma atenção especial, pois serão as mesmas em possíveis alterações de hardware e software.

Sobre o interpretador Basic em ROM são fornecidas descrições detalhadas de suas operações no capítulo 5.

Para aqueles que desejarem "desassemblar" as posições da ROM, encontrarão, neste capítulo, as áreas de dados que não contêm instruções. Uma listagem completa consome, aproximadamente, 400 páginas.

No capítulo 6 temos uma análise do mapeamento da memória, onde foram relacionadas as variáveis do BIOS/interpretador, na forma de linguagem Assembler. No final do capítulo são apresentados os ganchos, no total de 112, com seus respectivos endereços de chamada e comentário sobre a sua função.

No último capítulo são apresentados alguns programas em Assembler como: matriz do teclado, texto gráfico de 40 colunas, "Bubble Sort" de strings, dump de tela gráfica, editor de caracteres, etc.

O livro possui índice analítico que facilita a localização de uma informação específica sobre um determinado assunto.

Cartas

Foi com grata satisfação que recebi a notícia da publicação de uma nova revista destinada à linha MSX.

Aproveitando, gostaria de enviar algumas dúvidas e, se fosse possível, publicá-las no próximo número da revista CPU.

1º) É possível transformar o MSX 1.1 em MSX 2.0? Como?

2º) É possível aumentar a memória RAM trocando CIs por outros? Quais?

3º) Como posso usar um Drive de 5 1/4" e um de 3 1/2" simultaneamente?

Alcioni Nelson Silveira
Caixa Postal 185
Francisco Beltrão – PR

RESPOSTA:

Prezado Alcioni,

Respondemos, abaixo, às suas perguntas.

1º) A adaptação de um MSX 1.1 em 2.0 é economicamente inviável, pois teríamos que fazê-la através dos slots e o circuito seria por demais complexo. O melhor seria aguardar o lançamento do MSX 2.0 no Brasil.

2º) Não podemos, simplesmente, trocar os integrados de memória, pois outros tipos de maior capacidade têm pinagem e sistema de endereçamento diferentes. Expansões de memória somente através dos slots.

3º) Se a interface for capaz de formatar discos de 5 1/4" e 3 1/2", os dois tipos podem ser utilizados simultaneamente. Para maiores informações sugiro a leitura do livro "Drives Leopard 3 1/2" – Novos horizontes para o seu MSX", da Editora Aleph.

MATEMÁGICA

J. L. FONSECA

Na coluna deste mês, vamos nos dedicar a uma área que é uma das mais fascinantes dentro da matemática recreativa: a criptografia.

A criptografia é uma atividade tão antiga quanto a necessidade de transmitir mensagens secretas com segurança. No entanto, só com o advento dos computadores ela atingiu o destaque e a sofisticação de que goza hoje em dia.

Embora os melhores sistemas de hoje sejam implementados em computadores caros ou até em circuitos eletrônicos dedicados, o seu MSX pode ser de grande valia no estudo e uso das técnicas mais simples ou quando a velocidade não é um fator primordial.

Vamos, pois, apresentar aqui algumas técnicas básicas de cifragem, isto é, a transformação de uma mensagem em linguagem comum para uma já codificada segundo um código pré-estabelecido.

O primeiro método que vamos ver é o da listagem 1. Este método, um dos mais antigos, é conhecido como a cifra de Cesar, pois foi usado pelo mesmo nas comunicações com as suas legiões. O mesmo consiste em substituir cada letra por uma outra, geralmente dentro de uma seqüência que facilite o processo de codificação e decodificação.

Na forma aqui implementada, o programa pede uma mensagem a codificar ou decodificar, a qual será guardada em A$, e um deslocamento, que pode variar de 0 a 127. Ele pergunta, também, se você deseja codificar ou decodificar a mensagem. O deslocamento é um número que define como os caracteres serão substituídos. Por exemplo, se o deslocamento for igual a 3, a letra "a" será substituída pela letra "d", a "b" pela "e", e assim por diante. Na decodificação usamos o processo inverso.

Este método sofre a desvantagem de ser facilmente decifrável por quem não conhece o deslocamento, o que o torna pouco seguro e, por isso, desaconselhável para codificar mensagens com informações mais sigilosas.

Uma família de técnicas mais avançadas e mais seguras usa uma propriedade de função XOR, que faz com que um valor passado a esta função por duas vezes, com o mesmo argumento, não se altere. Para exemplificá-las, damos a listagem 2 que é um dos modos menos sofisticados de usar esta função na criptografia, mas que, por ser simples, exemplifica bem o uso da função.

Este programa pede a mensagem a codificar ou decodificar que será guardada em A$ e a chave que será guardada em C$. Na hora de imprimir o resultado, podem acontecer coisas estranhas, tais como não aparecer nada ou limpar a tela. Isto se deve ao fato de que os caracteres gerados podem ser códigos de controle. No entanto, quando usado em arquivos, este fato não cria problemas e é até útil, pois dificulta a decodificação dos mesmos.

Por hoje, não vamos nos alongar mais, mas voltaremos a este tema fascinante em colunas posteriores.

Até à próxima e bom divertimento.

```
40 'ROTINA 1
50 '
55 CLS
60 LINE INPUT "ENTRE COM A MENSAGEM: ";A$
70 INPUT "ENTRE COM O DESLOCAMENTO: ";DE
80 INPUT "CODIFICA OU DECODIFICA? C/D";C$
90 PRINT A$
100 B$=""
110 IF C$="D" OR C$="d" THEN GOTO 140
120 FOR I=1 TO LEN(A$): D$=MID$(A$,I,1):A=ASC(D$)+DE:B$=B$+CHR$(A):NEXT I
130 GOTO 150
140 FOR I=1 TO LEN(A$): D$=MID$(A$,I,1):A=ASC(D$)-DE:B$=B$+CHR$(A): NEXT I
150 PRINT B$
```

```
40 'ROTINA 2
50 '
55 CLS
60 LINE INPUT "ENTRE COM A MENSGEM: ";A$
70 LINE INPUT "ENTRE COM A CHAVE: ";C$
80 PRINT A$
90 S$=""
100 FOR I=1 TO LEN(A$): J=(I MOD LEN(C$))+1: D$=MID$(A$,I,1):E$=MID$(C$,J,1):S$=S$+CHR$(ASC(D$) XOR ASC(E$)): NEXT I
110 PRINT S$
```

JOGOS & HIGH SCORES

Jogo	Pontuação	Estágio
Alien 8	49%	
Alpha Blaster	89.235	
Barnstormer	279.855	12
Beamrider	133.380	25
Blagger	231.520	
Boom	99.240	34
Boulderdash	59.848	F/4
Bounder	286.726	5
Boxing		10
Buck Rogers	310.900	6
Centipede	53.795	7
Chiller	33.481	
Choro Q	42.380	
Circus Charlie	1.198.460	105
Disk Warrior	1.400.000	
Dogfighter	10.100	
Elidom	94%	
Eric & Floaters	1.844.160	
Finders Keepers	18.323	
Fire Rescue	29.540	
Flight Deck	6.410	
Fruit Frank	21.000	
Galaga	244.010	
Ghostbusters	$999.900	
Golf	28	
Gridtrap	558.120	
Gunfright	$150.000	51
Heist	384.201	
Hero	692.120	Pro
Highway	339.360	4
Hopper	100.050	3
Hotshoe	187.575	19
Hunchback	2.700.000	
Hyper Rally	239.500	
Hyper Sports I	2.050.800	51
Hyper Sports II	500.500	
Hyper Sports III	59.713	
Hyper Viper	127.500	
Karate	999.999	
Jet Fighter	214.590	
Jet Set Willy II	120	
King's Valley	5.642.600	928
Knightmare	238.020	
Lazy Jones	149.650	
Les Flics	100.200	
Le Mans	42.530	8
Manic Miner	117.321	52
Máxima	211.120	120
Monkey Academy	305.300	
Mopiranger	620.400	44
Mutant	737	7
Nightshade	137.000	13%
Ninja I	23.550	
Oh Mummy	5.030	
Oh No!	76.250	
Oil's Well	198.400	
Panic Junction	14.919	10
Pastfinder	24.205	
Pillbox	2.800	3
Pinball	1.240.680	
Pitfall II	199.000	
Polar Star	289.990	8
Punchy	6.959.870	
Price Magik	12%	18
Pyramid Warp	820.758	
River Raid	73.450	38
Road Fighter	998.675	7
Roller Ball	3.120.180	
Scion	67.900	
Soccer	40-0	
Spooks & Ladders	189.930	
Step Up	60.250	
Stop the Express	7.360	2/2
Super Cobra	501.100	62
Sweet Acorn	6.348.460	240
Tennis	6-0 6-0	
The Wreck	23.975	
Time Bandits	9.990	8
Time Curb	176.050	
Time Pilot	689.000	
Turmoil	11.740	
Vacumania	22.340	
Valkyr	23.975	

Se você já obteve um high score mais alto dos aqui apresentados, ou em qualquer outro jogo, envie-nos sua pontuação acompanhada de alguma comprovação, como fotografia da tela ou descrição das fases percorridas, para que possamos publicá-la, juntamente com o seu nome.

Se você é fera, nada mais justo do que o seu nome constar na seção de High Scores de CPU.

Os jogos que oferecem facilidades adicionais, com tiro múltiplo, vidas infinitas, etc., só serão considerados na sua versão original.

MSX CLUBE

THE MSX CLUB (Wales)
C/O PJ Morgan
230 Dunvant Road
Swansea
West Glamorgan SA2
7SR

MSX West
C/O Mark Smith
14 Beach Hill
Wellington
Somerset

The MSX Computer Club
C/O Dean Adams
173 Hampden Way
Southgate
London N14

MSX User's Club
C/O V. W. Warren
32 Stafford Road
Great Yarmouth
NR31 OEX

Yamaha DX/MSX Users Club
C/O Tony Wride
PO Box 6
Ripon
North Yorks HG4 2Qt

MSX User Group
C/O Andrew Phillips
Room 5
14 Moor Street
Omskirk
Lancashire

Caso você deseje formar um clube de usuários da linha MSX, escreva-nos fornecendo-nos o seu nome, nome do Clube e endereço para que possamos publicar em edições futuras.

KNIGHT TIME

Knight Time é um dos melhores adventures existentes para o MSX, sendo, na realidade, uma continuação do jogo Spellbound, onde você foi transportado para bordo de uma nave no século XXV e para bordo de uma nave no século XXV e irá tentar voltar para o seu tempo, convencendo os Senhores do Tempo a mandá-lo de volta.

Estas dicas irão auxiliar, mas, antes, você deverá ter algum trabalho.

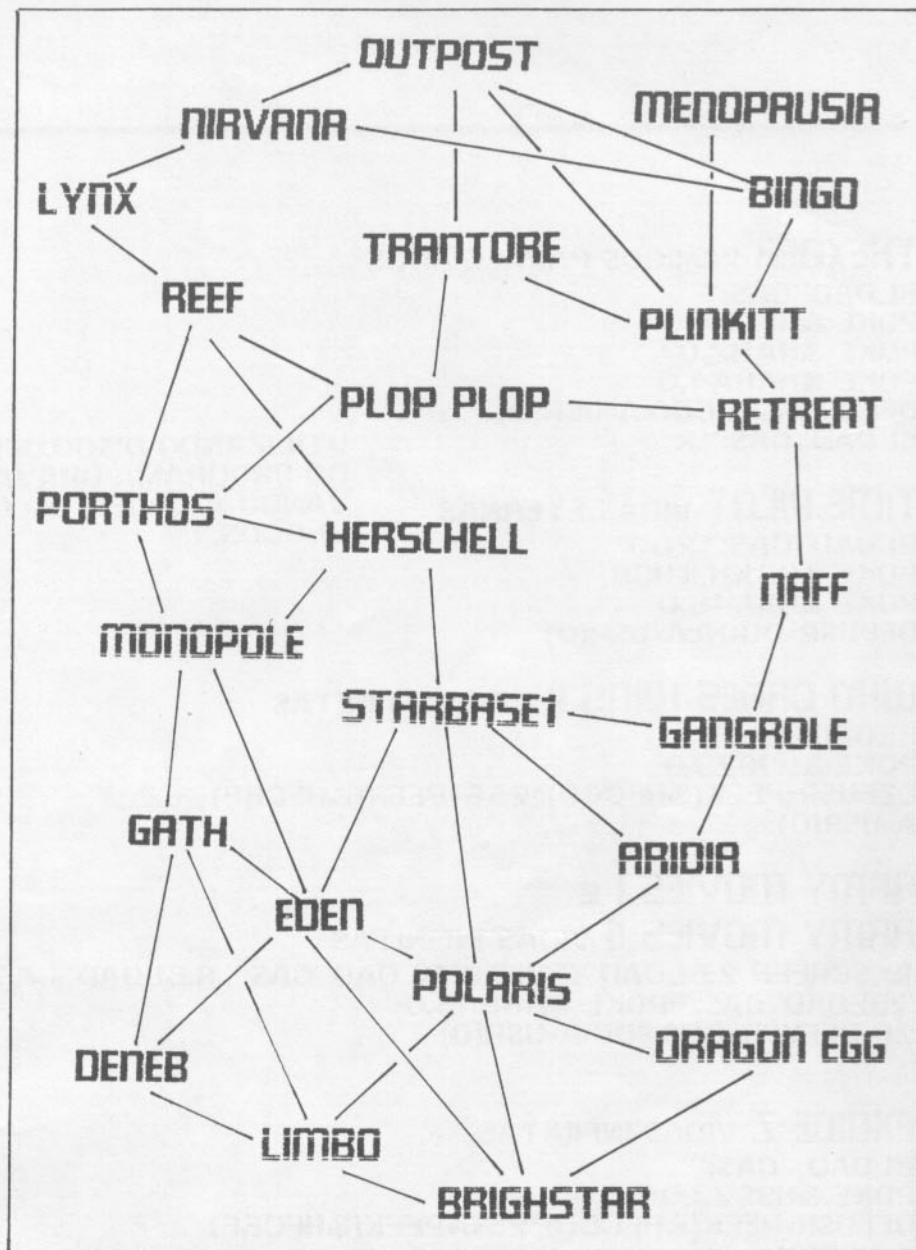
A primeira coisa a ser feita é retirar o manto de invisibilidade e soltar todas as coisas que estiver carregando em um lugar seguro. Feito isto, vá para a sala do transportador e peça ajuda ao Derby IV. Ele deixará cair um cartão de identificação que você deve pegar.

Para validar seu cartão de identificação, você deverá ir para a ponte e pegar os Mactablets do Sarab, os quais aumentarão a sua força. A seguir, pegue o filme instântaneo com o Gordon e vá para a sala de recreação. Apanhe a câmera e o filme. Encontre o S3E3, dê-lhe a câmera e o filme e peça-lhe ajuda. Ele irá tirar a sua fotografia. Pegue a fotografia e a cola. Se você fez tudo nesta ordem, a foto colará no cartão de identificação, o qual você poderá passar a utilizar.

Os personagens, agora, dar-lhe-ão atenção e você poderá comandar a nave estelar. Para fazer isto, você precisa do mapa estelar. Pegue o anúncio com o Sharon e deixe-o cair no local onde se encontra o mapa e a caneca do chopp. Você pode, agora, subir no anúncio e pegar o mapa. Pegue o mapa, a caneca e o anúncio e vá para a ponte. Dê o mapa estelar para o Gordon e você poderá dirigir a nave.

Você deve consertar a nave. Vá para a Starbase 1, via Polaris. Dê a caneca de chopp para o Gordon e peça-lhe ajuda. Ele irá consertar o transportador para você. A seguir, comunique-se com a Starbase 1, pedindo ajuda, e você receberá algumas coordenadas válidas. De posse das características, desça para a Starbase e inicie a exploração.

Use o mapa que fornecemos para marcar os seus movimentos pelos planetas.

Os pedaços do relógio do Sol deAlpha estão com Murphy e Hooper. Você irá encontrar Murphy no planeta Retreat e Hooper em Monopole. Os senhores do Sol poderão ser encontrados em Outpost.

Objetos úteis:

Bomba de quartzo	Explode planetas
Botas	Para passar os sistemas de segurança
Saco de batatas	Dê para Murphy
Máscara de gás	Protege-o contra gás
Espelho	Mostra o que você está carregando

Códigos do transportador

Coordenadas da nave	X0Y0Z0
Starbase	X1Y2Z3
Retreat	X8Y4Z1
Monopole	X1Y8Z4
Outpost	X8Y9Z6

DICAS

THEXDER INIMIGOS PARALIZADOS

```
BLOAD"CAS:"
POKE &H90E4,0
POKE &HA112,0
POKE &HABA4,0
DEFUSR=&HD000:A=USR(0)
BLOAD"CAS:",R
```

UTILIZANDO DISCO, MUDE O 'CAS' PELO NOME DO PROGRAMA GRAVADO NO DISCO, OBSERVANDO A ORDEM DO CARREGAMENTO DOS BLOCOS.

TIME PILOT VIDAS ETERNAS

```
BLOAD"CAS:",200
POKE &HC101,&HCB
POKE &H8A48,0
DEFUSR=D000:A=USR(0)
```

WHO DARES WINS II VIDAS INFINITAS

```
BLOAD"CAS:"
POKE &H9923,0
DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF)
A=USR(0)
```

ARMY MOVIES I e
ARMY MOVIES II VIDAS INFINITAS

```
10 SCREEN 2:BLOAD"CAS:",R"BLOAD"CAS:",R:BLOAD"CAS:
":BLOAD"CAS:":POKE &H88AB,0
20 DEFUSR=&H82DC:A=USR(0)
```

EXOIDE Z VIDAS INFINITAS

```
BLOAD "CAS:"
POKE &H9923,0
DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF)
A=USR(0)
```

GYRODINE IMUNIDADE TOTAL

```
BLOAD"CAS:"
POKE -25648,0
DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF)
A=USR(0)
BLOAD"CAS:",R
```

HYPER RALLY SEM INIMIGOS

```
BLOAD"CAS:"
POKE &H935B,0:POKE &H935C,1
DEFUSR=&HD000
A=USR(0)
```

TRICK BOY VARIAS BOLAS

```
BLOAD"CAS:"
POKE &H93AA,99
DEFUSR=PEEK(&HFCC0)*256+PEEK(&HFCBF)
A=USR(0)
```

Pura Tecnologia

DRIVE LEOPARD 3,5" PARA MSX*

Depois dele os outros vão ter que mudar.

O Leopard é o primeiro Drive nacional de 3,5". A mesma tecnologia utilizada em sua fabricação, foi transferida para o Conjunto Leopard para MSX.

Características:

- 500 Kb não formatados.
- A mais moderna Interface Controladora para MSX do mercado. Trabalha com o clock de 16 Mz (Padrão Mundial para MSX), e controla 2 Drives de 3,5" ou 5 1/4".
- Fonte Externa, ou seja, seus problemas de aquecimento estão definitivamente resolvidos.
- Os Drives de 3,5" são usados hoje por todos os grandes fabricantes de computadores pessoais do mundo.

Depois de tudo isso procure os nossos revendedores:

São Paulo: Audio - Amarosom - Bruno Blois e Cia. - Brenno Rossi - Cinótica - Fotóptica - Labracom - Mundisom - Plenisom - Shop Audio e Video - Pró-eletrônica **Belém:** Hot Club **Porto Alegre:** Brenno Rossi - Casa dos Gravadores - Cambial **Belo Horizonte:** Sleiman Programas e Sistemas **Curitiba:** Brenno Rossi - Opticas Boa Vista **Florianópolis:** Audio Center.

TECHNOAHEAD MAGNÉTICOS LTDA
Rua Visconde de Parnaíba, 2898 - fone (011) 264.5600 - SP

* M.R. GRADIENTE ELETRÔNICA